# AWS WAF and AWS Shield Advanced

## Developer Guide

## API Version 2015-08-24

# AWS WAF and AWS Shield Advanced: Developer Guide

# Table of Contents

# What Is AWS WAF and AWS Shield?

AWS WAF is a web application firewall that lets you monitor the HTTP and HTTPS requests that are forwarded to Amazon CloudFront or an Application Load Balancer. AWS WAF also lets you control access to your content. Based on conditions that you specify, such as the IP addresses that requests originate from or the values of query strings, CloudFront or an Application Load Balancer responds to requests either with the requested content or with an HTTP 403 status code (Forbidden). You also can configure CloudFront to return a custom error page when a request is blocked.

At the simplest level, AWS WAF lets you choose one of the following behaviors:

- **Allow all requests except the ones that you specify** – This is useful when you want CloudFront or an Application Load Balancer to serve content for a public website, but you also want to block requests from attackers.
- **Block all requests except the ones that you specify** – This is useful when you want to serve content for a restricted website whose users are readily identifiable by properties in web requests, such as the IP addresses that they use to browse to the website.
- **Count the requests that match the properties that you specify** – When you want to allow or block requests based on new properties in web requests, you first can configure AWS WAF to count the requests that match those properties without allowing or blocking those requests. This lets you confirm that you didn't accidentally configure AWS WAF to block all the traffic to your website. When you're confident that you specified the correct properties, you can change the behavior to allow or block requests.

Using AWS WAF has several benefits:

- Additional protection against web attacks using conditions that you specify. You can define conditions by using characteristics of web requests such as the following:
  - IP addresses that requests originate from.
  - Values in request headers.
  - Strings that appear in requests.
  - Length of requests.
  - Presence of SQL code that is likely to be malicious (known as *SQL injection*).
  - Presence of a script that is likely to be malicious (known as *cross-site scripting*).
- Rules that can allow, block, or count web requests that meet the specified conditions. Alternatively, rules can block or count web requests that not only meet the specified conditions, but also exceed a specified number of requests in any 5-minute period.
- Rules that you can reuse for multiple web applications.

- Real-time metrics and sampled web requests.
- Automated administration using the AWS WAF API.

# AWS Shield

You can use AWS WAF web access control lists (web ACLs) to help minimize the effects of a distributed denial of service (DDoS) attack. AWS also provides AWS Shield Standard and AWS Shield Advanced. AWS Shield Standard is automatically included at no extra cost beyond what you already pay for AWS WAF and your other AWS services. For added protection against DDoS attacks, AWS offers AWS Shield Advanced. AWS Shield Advanced provides expanded DDoS attack protection for your Elastic Load Balancing load balancers, CloudFront distributions, and Amazon Route 53 hosted zones.

For more information about AWS Shield Standard and AWS Shield Advanced, see AWS Shield (p. 82).

# Setting Up for AWS WAF and AWS Shield

This topic describes preliminary steps, such as creating an AWS account, to prepare you to use AWS WAF and AWS Shield Advanced. You do not get charged to set up this account and other preliminary items. You are only charged for AWS services that you use.

> **Note**
> AWS Shield Standard is included with AWS WAF and does not require additional setup. For more information, see How AWS Shield Works (p. 82).

Before you use AWS WAF or AWS Shield Advanced for the first time, complete the following tasks:

- Step 1: Sign Up for an AWS Account (p. 3)
- Step 2: Create an IAM User (p. 4)
- Step 3: Download Tools (p. 5)

## Step 1: Sign Up for an AWS Account

When you sign up for Amazon Web Services (AWS), your AWS account is automatically signed up for all services in AWS, including AWS WAF. You are charged only for the services that you use.

If you have an AWS account already, skip to the next task. If you don't have an AWS account, use the following procedure to create one.

### To sign up for AWS

1. Open https://aws.amazon.com/ and choose **Sign Up**.
2. Follow the on-page instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

Note your AWS account number, because you'll need it for the next task.

# Step 2: Create an IAM User

To use the AWS WAF console, you must sign in to confirm that you have permission to perform AWS WAF operations. You can use the root credentials for your AWS account, but we don't recommend it. For greater security and control of your account, we recommend that you use AWS Identity and Access Management (IAM) to do the following:

- Create an IAM user account for yourself or your business
- Either add the IAM user account to an IAM group that has administrative permissions, or grant the IAM user account administrative permissions directly

You then can sign in to the AWS WAF console (and other service consoles) by using a special URL and the credentials for the IAM user. You also can add other users to the IAM user account, and control their level of access to AWS services and to your resources.

> **Note**
> For information about creating access keys to access AWS WAF by using the AWS Command Line Interface (AWS CLI), Tools for Windows PowerShell, the AWS SDKs, or the AWS WAF API, see Managing Access Keys for IAM Users.

If you signed up for AWS but have not created an IAM user for yourself, you can create one using the IAM console. If you aren't familiar with using the console, see Working with the AWS Management Console for an overview.

### To create an IAM user for yourself and add the user to an Administrators group

1. Sign in to the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose **Users**, and then choose **Add user**.
3. For **User name**, type a user name, such as `Administrator`. The name can consist of letters, digits, and the following characters: plus (+), equal (=), comma (,), period (.), at (@), underscore (_), and hyphen (-). The name is not case sensitive and can be a maximum of 64 characters in length.
4. Select the check box next to **AWS Management Console access**, select **Custom password**, and then type the new user's password in the text box. You can optionally select **Require password reset** to force the user to select a new password the next time the user signs in.
5. Choose **Next: Permissions**.
6. On the **Set permissions for user** page, choose **Add user to group**.
7. Choose **Create group**.
8. In the **Create group** dialog box, type the name for the new group. The name can consist of letters, digits, and the following characters: plus (+), equal (=), comma (,), period (.), at (@), underscore (_), and hyphen (-). The name is not case sensitive and can be a maximum of 128 characters in length.
9. For **Filter**, choose **Job function**.
10. In the policy list, select the check box for **AdministratorAccess**. Then choose **Create group**.
11. Back in the list of groups, select the check box for your new group. Choose **Refresh** if necessary to see the group in the list.
12. Choose **Next: Review** to see the list of group memberships to be added to the new user. When you are ready to proceed, choose **Create user**.

You can use this same process to create more groups and users, and to give your users access to your AWS account resources. To learn about using policies to restrict users' permissions to specific AWS resources, go to Access Management and Example Policies for Administering AWS Resources.

To sign in as this new IAM user, first sign out of the AWS console. Then use the following URL, where *your_aws_account_id* is your AWS account number without the hyphens. For example, if your AWS account number is `1234-5678-9012`, your AWS account ID is `123456789012`:

```
https://your_aws_account_id.signin.aws.amazon.com/console/
```

Enter the IAM user name and password that you just created. When you're signed in, the navigation bar displays "*your_user_name @ your_aws_account_id*".

If you don't want the URL for your sign-in page to contain your AWS account ID, you can create an account alias. From the IAM dashboard, choose **Customize** and enter an alias, such as your company name. To sign in after you create an account alias, use the following URL:

```
https://your_account_alias.signin.aws.amazon.com/console/
```

To verify the sign-in link for IAM users for your account, open the IAM console and check under the **IAM users sign-in link** on the dashboard.

# Step 3: Download Tools

The AWS Management Console includes a console for AWS WAF, but if you want to access AWS WAF programmatically, the following documentation and tools will help you:

- If you want to call the AWS WAF API without having to handle low-level details like assembling raw HTTP requests, you can use an AWS SDK. The AWS SDKs provide functions and data types that encapsulate the functionality of AWS WAF and other AWS services. To download an AWS SDK, see the applicable page, which also includes prerequisites and installation instructions:
  - Java
  - JavaScript
  - .NET
  - Node.js
  - PHP
  - Python
  - Ruby

  For a complete list of AWS SDKs, see Tools for Amazon Web Services.
- If you're using a programming language for which AWS doesn't provide an SDK, the AWS WAF API Reference documents the operations that AWS WAF supports.
- The AWS Command Line Interface (AWS CLI) supports AWS WAF. The AWS CLI lets you control multiple AWS services from the command line and automate them through scripts. For more information, see AWS Command Line Interface.
- AWS Tools for Windows PowerShell supports AWS WAF. For more information, see AWS Tools for Windows PowerShell Reference.

# AWS WAF

AWS WAF is a web application firewall that lets you monitor the HTTP and HTTPS requests that are forwarded to Amazon CloudFront or an Application Load Balancer. AWS WAF also lets you control access to your content. Based on conditions that you specify, such as the IP addresses that requests originate from or the values of query strings, CloudFront or an Application Load Balancer responds to requests either with the requested content or with an HTTP 403 status code (Forbidden). You also can configure CloudFront to return a custom error page when a request is blocked.

Topics

## How AWS WAF Works

You use AWS WAF to control how Amazon CloudFront or an Application Load Balancer responds to web requests. You start by creating conditions, rules, and web access control lists (web ACLs). You define your conditions, combine your conditions into rules, and combine the rules into a web ACL.

**Conditions**

Conditions define the basic characteristics that you want AWS WAF to watch for in web requests:

- Scripts that are likely to be malicious. Attackers embed scripts that can exploit vulnerabilities in web applications. This is known as *cross-site scripting*.
- IP addresses or address ranges that requests originate from.
- Length of specified parts of the request, such as the query string.

- SQL code that is likely to be malicious. Attackers try to extract data from your database by embedding malicious SQL code in a web request. This is known as *SQL injection*.
- Strings that appear in the request, for example, values that appear in the `User-Agent` header or text strings that appear in the query string.

Some conditions take multiple values. For example, you can specify up to 10,000 IP addresses or IP address ranges in an IP condition.

**Rules**

You combine conditions into rules to precisely target the requests that you want to allow, block, or count. AWS WAF provides two types of rules:

**Regular rule**

Regular rules use only conditions to target specific requests. For example, based on recent requests that you've seen from an attacker, you might create a rule that includes the following conditions:

- The requests come from 192.0.2.44.
- They contain the value `BadBot` in the `User-Agent` header.
- They appear to include SQL-like code in the query string.

When a rule includes multiple conditions, as in this example, AWS WAF looks for requests that match all conditions—that is, it `AND`s the conditions together.

**Rate-based rule**

Rate-based rules are similar to regular rules, with one addition: a rate limit. Rate-based rules count the requests that arrive from a specified IP address every five minutes. The rule can trigger an action if the number of requests exceed the rate limit.

You can combine conditions with the rate limit. In this case, if the requests match all of the conditions and the number of requests exceed the rate limit in any five-minute period, the rule will trigger the action designated in the web ACL.

For example, based on recent requests that you've seen from an attacker, you might create a rate-based rule that includes the following conditions:

- The requests come from 192.0.2.44.
- They contain the value `BadBot` in the `User-Agent` header.

In this rate-based rule, you also define a rate limit. In this example, let's say that you create a rate limit of 15,000. Requests that meet both of the preceding conditions and exceed 15,000 requests per five minutes trigger the rule's action (block or count), which is defined in the web ACL.

Requests that do not meet both conditions will not be counted towards the rate limit and would not be blocked by this rule.

As a second example, suppose that you want to limit requests to a particular page on your website. To do this, you could add the following string match condition to a rate-based rule:

- The **Part of the request to filter on** is `URI`.
- The **Match Type** is `Starts with`.
- A **Value to match** is `login`.

Further, you specify a `RateLimit` of 15,000.

By adding this rate-based rule to a web ACL, you could limit requests to your login page without affecting the rest of your site.

**Important**

You should add at least one condition to a regular rule. A regular rule with no conditions will not match any requests and therefore the rule's action (allow, count, block) will never be triggered.

However, conditions are optional for rate-based rules. If you don't add any conditions to a rate-based rule, AWS WAF assumes that *all* requests match the rule and therefore will be counted against the rate limit when arriving from the same IP address. Requests from the same IP address that exceed the rate limit will trigger the rule's action (count or block).

**Web ACLs**

After you combine your conditions into rules, you combine the rules into a web ACL. This is where you define an action for each rule—allow, block, or count—and a default action:

**An action for each rule**

When a web request matches all the conditions in a rule, AWS WAF can either block the request or allow the request to be forwarded to CloudFront or an Application Load Balancer. You specify the action that you want AWS WAF to perform for each rule.

AWS WAF compares a request with the rules in a web ACL in the order in which you listed the rules. AWS WAF then takes the action that is associated with the first rule that the request matches. For example, if a web request matches one rule that allows requests and another rule that blocks requests, AWS WAF will either allow or block the request depending on which rule is listed first.

If you want to test a new rule before you start using it, you also can configure AWS WAF to count the requests that meet all the conditions in the rule. As with rules that allow or block requests, a rule that counts requests is affected by its position in the list of rules in the web ACL. For example, if a web request matches a rule that allows requests and another rule that counts requests, and if the rule that allows requests is listed first, the request isn't counted.

**A default action**

The default action determines whether AWS WAF allows or blocks a request that doesn't match all the conditions in any of the rules in the web ACL. For example, suppose you create a web ACL and add only the rule that you defined before:

- The requests come from 192.0.2.44.
- They contain the value `BadBot` in the `User-Agent` header.
- They appear to include malicious SQL code in the query string.

If a request doesn't meet all three conditions in the rule and if the default action is `ALLOW`, AWS WAF forwards the request to CloudFront or an Application Load Balancer, and the service responds with the requested object.

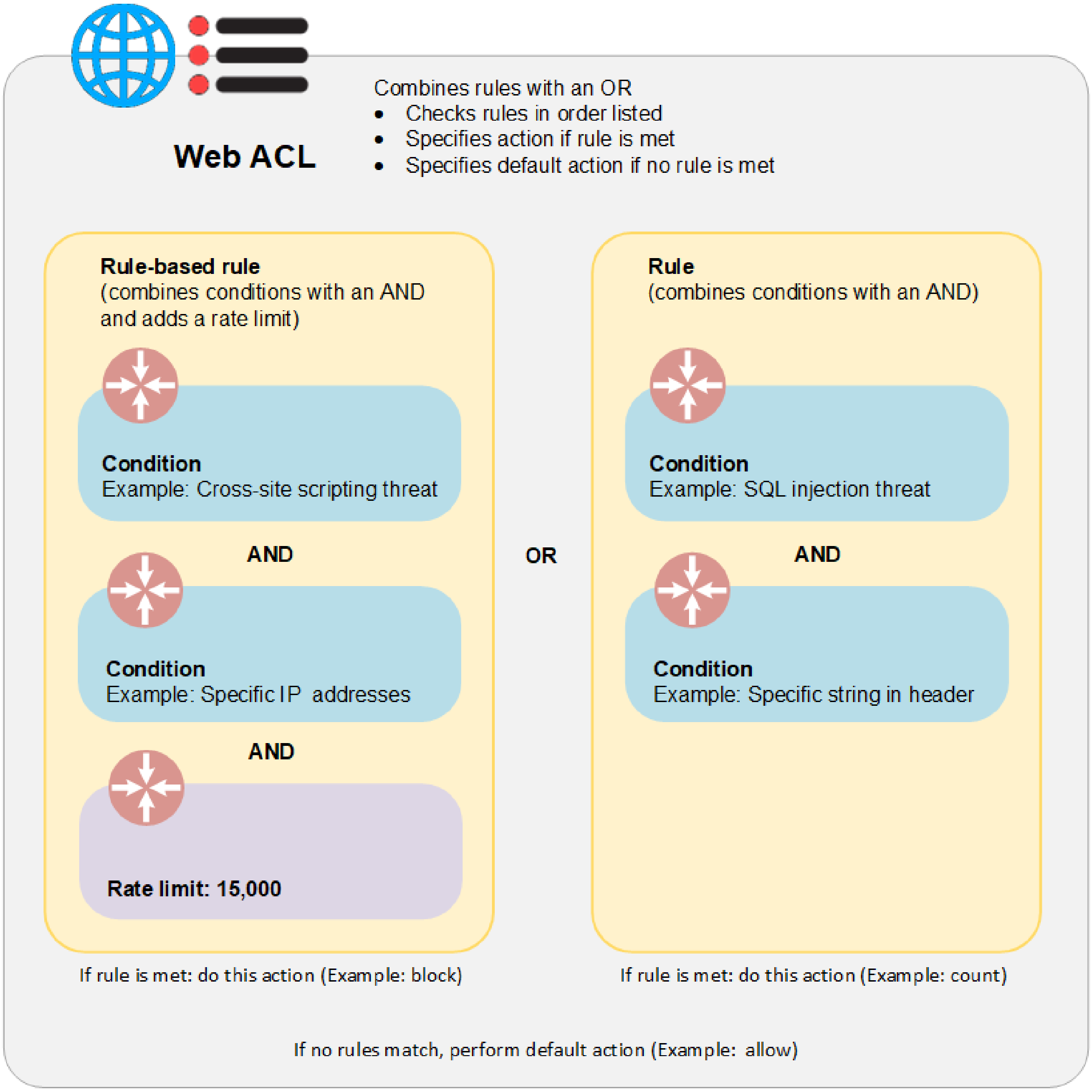
If you add two or more rules to a web ACL, AWS WAF performs the default action only if a request doesn't satisfy all the conditions in any of the rules. For example, suppose you add a second rule that contains one condition:

- Requests that contain the value `BIGBadBot` in the `User-Agent` header.

AWS WAF performs the default action only when a request doesn't meet all three conditions in the first rule and doesn't meet the one condition in the second rule.

On rare occasions, AWS WAF might encounter an internal error that delays the response to CloudFront or an Application Load Balancer about whether to allow or block a request. On those occasions, CloudFront or an Application Load Balancer typically will serve the content.

The following illustration shows how AWS WAF checks the rules and performs the actions based on those rules.


Web ACL
Combines rules with an OR
• Checks rules in order listed
• Specifies action if rule is met
• Specifies default action if no rule is met
Rule-based rule
(combines conditions with an AND and adds a rate limit)
Condition
Example: Cross-site scripting threat
AND
Condition
Example: Specific IP addresses
AND
Rate limit: 15,000
OR
Rule
(combines conditions with an AND)
Condition
Example: SQL injection threat
AND
Condition
Example: Specific string in header
If rule is met: do this action (Example: block)
If rule is met: do this action (Example: count)
If no rules match, perform default action (Example: allow)

# AWS WAF Pricing

With AWS WAF, you pay only for the web ACLs and rules that you create, and for the number of HTTP requests that AWS WAF inspects. For more information, see AWS WAF Pricing.

# Getting Started with AWS WAF

This tutorial shows how to use AWS WAF to perform the following tasks:

- Set up AWS WAF.

- Create a web access control list (web ACL) using the AWS WAF console, and specify the conditions that you want to use to filter web requests. For example, you can specify the IP addresses that the requests originate from and values in the request that are used only by attackers.
- Add the conditions to a rule. Rules let you target the web requests that you want to block or allow. A web request must match all the conditions in a rule before AWS WAF blocks or allows requests based on the conditions that you specify.
- Add the rules to your web ACL. This is where you specify whether you want to block web requests or allow them based on the conditions that you add to each rule.
- Specify a default action, either block or allow. This is the action that AWS WAF takes when a web request doesn't match any of your rules.
- Choose the Amazon CloudFront distribution that you want AWS WAF to inspect web requests for. This tutorial covers the steps only for CloudFront, but the process for an Application Load Balancer essentially is the same. AWS WAF for CloudFront is available for all regions. AWS WAF for use with an Application Load Balancer is available for the following regions:
  - US East (N. Virginia)
  - US West (Oregon)
  - EU (Ireland)
  - Asia Pacific (Tokyo)

> **Note**
> AWS typically bills you less than US $0.25 per day for the resources that you create during this tutorial. When you're finished with the tutorial, we recommend that you delete the resources to prevent incurring unnecessary charges.

Topics

## Step 1: Set Up AWS WAF

If you already signed up for an AWS account and created an IAM user as described in Setting Up for AWS WAF and AWS Shield (p. 3), go to Step 2: Create a Web ACL (p. 10).

If not, go to Setting Up for AWS WAF and AWS Shield (p. 3) and perform at least the first two steps. (You can skip downloading tools for now because this Getting Started topic focuses on using the AWS WAF console.)

## Step 2: Create a Web ACL

The AWS WAF console guides you through the process of configuring AWS WAF to block or allow web requests based on conditions that you specify, such as the IP addresses that the requests originate from or values in the requests. In this step, you create a web ACL.

### To create a web ACL

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. If this is your first time using AWS WAF, choose **Go to AWS WAF**, and then choose **Configure web ACL**.

   If you've used AWS WAF before, choose **Web ACLs** in the navigation pane, and then choose **Create web ACL**.
3. On the **Name web ACL** page, for **Web ACL name**, type a name.

   **Note**
   You can't change the name after you create the web ACL.
4. For **CloudWatch metric name**, type a name. The name can contain only alphanumeric characters (A-Z, a-z, 0-9). It can't contain whitespace.

   **Note**
   You can't change the name after you create the web ACL.
5. For **Region**, choose a region. If you will associate this web ACL with a CloudFront distribution, choose **Global (CloudFront)**.
6. For **AWS resource to associate**, choose the resource that you want to associate with your web ACL, and then choose **Next**.

# Step 3: Create an IP Match Condition

An IP match condition specifies the IP addresses or IP address ranges that requests originate from. In this step, you create an IP match condition. In a later step, you specify whether you want to allow requests or block requests that originate from the specified IP addresses.

> **Note**
> For more information about IP match conditions, see Working with IP Match Conditions (p. 35).

### To create an IP match condition

1. On the **Create conditions** page, for **IP match conditions**, choose **Create condition**.
2. In the **Create IP match condition** dialog box, for **Name**, type a name. The name can contain only alphanumeric characters.
3. For **Address**, type **192.0.2.0/24**. This IP address range, specified in CIDR notation, includes the IP addresses from 192.0.2.0 to 192.0.2.255. (The 192.0.2.0/24 IP address range is reserved for examples, so no web requests will originate from these IP addresses.)

   You can specify /8, /16, /24, and /32 IP address ranges. (To specify a single IP address, such as 192.0.2.44, type **192.0.2.44/32**.) Other ranges aren't supported.

   For more information about CIDR notation, see the Wikipedia article Classless Inter-Domain Routing.
4. Choose **Create**.

# Step 4: Create a String Match Condition

A string match condition identifies the strings that you want AWS WAF to search for in a request, such as a specified value in a header or in a query string. Usually, a string consists of printable ASCII characters, but you can specify any character from hexadecimal 0x00 to 0xFF (decimal 0 to 255). In this step, you create a string match condition. In a later step, you specify whether you want to allow or block requests that contain the specified strings.

**Note**
For more information about string match conditions, see Working with String Match Conditions (p. 45).

### To create a string match condition

1. On the **Create conditions** page, for **String match conditions**, choose **Create condition**.
2. In the **Create string match condition** dialog box, type the following values:

   **Name**

   Type a name. The name can contain only alphanumeric characters.

   **Part of the request to filter on**

   Choose the part of the web request that you want AWS WAF to inspect for a specified string.

   For this example, choose **Header**.

   **Note**
   If you choose **Body** for the value of **Part of the request to filter on**, AWS WAF inspects only the first 8192 bytes (8 KB) because CloudFront forwards only the first 8192 bytes for inspection. To allow or block requests for which the body is longer than 8192 bytes, you can create a size constraint condition. (AWS WAF gets the length of the body from the request headers.) For more information, see Working with Size Constraint Conditions (p. 37).

   **Header (Required if "Part of the request to filter on" is "Header")**

   Because you chose **Header** for **Part of the request to filter on**, you must specify which header you want AWS WAF to inspect. Type **User-Agent**. (This value is not case sensitive.)

   **Match type**

   Choose where the specified string must appear in the **User-Agent** header, for example, at the beginning, at the end, or anywhere in the string.

   For this example, choose **Exactly matches**, which indicates that AWS WAF inspects web requests for a header value that is identical to the value that you specify.

   **Transformation**

   In an effort to bypass AWS WAF, attackers use unusual formatting in web requests, for example, by adding whitespace or by URL-encoding some or all of the request. Transformations convert the web request to a more standard format by removing whitespace, by URL-decoding the request, or by performing other operations that eliminate much of the unusual formatting that attackers commonly use.

   For this example, choose **None**.

   **Value is base64 encoded**

   When the value that you type in **Value to match** is already base64-encoded, select this check box.

   For this example, don't select the check box.

   **Value to match**

   Specify the value that you want AWS WAF to search for in the part of web requests that you indicated in **Part of the request to filter on**. Use the format that you specified in **String format**.

   For this example, type **BadBot**. AWS WAF will inspect the `User-Agent` header in web requests for the value **BadBot**.

The maximum length of **Value to match** is 50 bytes. If you want to specify a base64-encoded value, the limit is 50 bytes before encoding.

3. If you want AWS WAF to inspect web requests for multiple values, such as a `User-Agent` header that contains `BadBot` and a query string that contains `BadParameter`, you have two choices:

   - If you want to allow or block web requests only when they contain both values (`AND`), you create one string match condition for each value.
   - If you want to allow or block web requests when they contain either value or both (`OR`), you add both values to the same string match condition.

   For this example, choose **Create**.

## Step 5: Create a SQL Injection Match Condition

A SQL injection match condition identifies the part of web requests, such as a header or a query string, that you want AWS WAF to inspect for malicious SQL code. Attackers use SQL queries to extract data from your database. In this step, you create a SQL injection match condition. In a later step, you specify whether you want to allow requests or block requests that appear to contain malicious SQL code.

> **Note**
> For more information about string match conditions, see Working with SQL Injection Match Conditions (p. 42).

### To create a SQL injection match condition

1. On the **Create conditions** page, for **SQL injection match conditions**, choose **Create condition**.
2. In the **Create SQL injection match condition** dialog box, type the following values:

   **Name**

   Type a name.

   **Part of the request to filter on**

   Choose the part of web requests that you want AWS WAF to inspect for malicious SQL code.

   For this example, choose **Query string**.

   > **Note**
   > If you choose **Body** for the value of **Part of the request to filter on**, AWS WAF inspects only the first 8192 bytes (8 KB) because CloudFront forwards only the first 8192 bytes for inspection. To allow or block requests for which the body is longer than 8192 bytes, you can create a size constraint condition. (AWS WAF gets the length of the body from the request headers.) For more information, see Working with Size Constraint Conditions (p. 37).

   **Transformation**

   For this example, choose **URL decode**.

   Attackers use unusual formatting, such as URL encoding, in an effort to bypass AWS WAF. The **URL decode** option eliminates some of that formatting in the web request before AWS WAF inspects the request.
3. Choose **Create**.
4. Choose **Next**.

# Step 6: (Optional) Create Additional Conditions

AWS WAF includes other conditions, including the following:

- **Size constraint conditions** – Identifies the part of web requests, such as a header or a query string, that you want AWS WAF to check for length. For more information, see Working with Size Constraint Conditions (p. 37).
- **Cross-site scripting match conditions** – Identifies the part of web requests, such as a header or a query string, that you want AWS WAF to inspect for malicious scripts. For more information, see Working with Cross-site Scripting Match Conditions (p. 31).

You can optionally create these conditions now, or you can skip to Step 7: Create a Rule and Add Conditions (p. 14).

# Step 7: Create a Rule and Add Conditions

You create a rule to specify the conditions that you want AWS WAF to search for in web requests. If you add more than one condition to a rule, a web request must match all the conditions in the rule for AWS WAF to allow or block requests based on that rule.

> **Note**
> For more information about rules, see Working with Rules (p. 50).

### To create a rule and add conditions

1. On the **Create rules** page, choose **Create rule**.
2. In the **Create rule** dialog box, type the following values:

   **Name**

   Type a name.

   **CloudWatch metric name**

   Type a name for the CloudWatch metric that AWS WAF will create and will associate with the rule. The name can contain only alphanumeric characters (A-Z, a-z, 0-9). It can't contain whitespace.

   **Rule type**

   Choose either `Regular rule` or `Rate based rule`. Rate based rules are identical to regular rules but also take into account how many requests arrive from the identified IP address every five minutes. For more information on these rule types, see How AWS WAF Works (p. 6).

   **Rate limit**

   If you are creating a rate-based rule, enter the maximum number of requests from a single IP address allowed in a five-minute period.
3. For the first condition that you want to add to the rule, specify the following settings:

   - Choose whether you want AWS WAF to allow or block requests based on whether a web request does or does not match the settings in the condition.

     For this example, choose **does**.
   - Choose the type of condition that you want to add to the rule: an IP match set condition, a string match set condition, or a SQL injection match set condition.

     For this example, choose **originate from IP addresses in**.

- Choose the condition that you want to add to the rule.

  For this example, choose the IP match condition that you created in previous tasks.

4. Choose **Add another condition**.
5. Add the string match condition that you created earlier. Specify the following values:
   - **When a request does**
   - **match at least one of the filters in the string match condition**
   - Choose your string match condition.
6. Choose **Add another condition**.
7. Add the SQL injection match condition that you created earlier. Specify the following values:
   - **When a request does**
   - **match at least one of the filters in the SQL injection match condition**
   - Choose your SQL injection match condition.
8. Choose **Add another condition**.
9. Add the size constraint condition that you created earlier. Specify the following values:
   - **When a request does**
   - **match at least one of the filters in the size constraint condition**
   - Choose your size constraint condition.
10. Choose **Review and create**.

# Step 8: Add the Rule to a Web ACL

When you add the rule to a web ACL, you specify the following settings:

- The action that you want AWS WAF to take on web requests that match all the conditions in the rule: allow, block, or count the requests.
- The default action for the web ACL. This is the action that you want AWS WAF to take on web requests that *do not* match all the conditions in the rule: allow or block the requests.

You can add more than one rule to a web ACL. If a web ACL contains more than one rule, AWS WAF inspects web requests based on the conditions in each rule in the order that the rules are listed. For example, if a web request matches one rule that allows requests and another rule that blocks requests, AWS WAF either allows or blocks the request depending on which rule is listed first.

> **Note**
> For more information about rules, see Working with Web ACLs (p. 53). For a list of limits on AWS WAF objects, such as the number of rules that you can create per AWS account, see AWS WAF Limits (p. 80).

### To add a rule to a web ACL

1. In the navigation pane, choose **Create rules**. Under **Add rules to a web ACL**, the rule that you created in a preceding task is already selected because it's the only rule you have.

   When you have more than one rule, you choose the rule that you want to add from the Rules list, and then choose **Add rule to web ACL**.
2. For the rule that you added to the web ACL in the preceding step, choose whether you want AWS WAF to allow, block, or count requests that match all the conditions in the rule.

   For this example, choose **Block**.

3. Choose the default action for the web ACL. AWS WAF takes this action on web requests that do not match all the conditions in the rule.

   For this example, choose **Allow all requests that don't match any rules**.
4. Choose **Review and create**.
5. Confirm the summary information, and then choose **Confirm and create**.

AWS WAF starts blocking CloudFront web requests that match all the following conditions:

- The value of the `User-Agent` header is `BadBot`
- The requests originate from IP addresses in the range 192.0.2.0-192.0.2.255
- The requests appear to include malicious SQL code in the query string

AWS WAF allows CloudFront to respond to any requests that don't meet all three of these conditions.

# Step 9: Clean Up Your Resources

You've now successfully completed the tutorial. To prevent your account from accruing additional AWS WAF charges, you should clean up the AWS WAF objects that you created. Alternatively, you can change the configuration to match the web requests that you really want to allow, block, and count.

> **Note**
> AWS typically bills you less than US $0.25 per day for the resources that you create during this tutorial. When you're finished, we recommend that you delete the resources to prevent incurring unnecessary charges.

### To delete the objects that AWS WAF charges for

1. Disassociate your web ACL from your CloudFront distribution:

   a. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
   b. Choose the web ACL that you want to delete.
   c. In the right pane, on the **Rules** tab, go to the **AWS resources using this web ACL** section. For the CloudFront distribution that you associated the web ACL with, choose the **x** in the **Type** column.
2. Remove the conditions from your rule:

   a. In the navigation pane, choose **Rules**.
   b. Choose the rule that you created during the tutorial.
   c. Choose **Edit rule**.
   d. Choose the **x** at the right of each condition heading.
   e. Choose **Update**.
3. Remove the rule from your web ACL, and delete the web ACL:

   a. In the navigation pane, choose **Web ACLs**.
   b. Choose the web ACL that you created during the tutorial.
   c. On the **Rules** tab, choose **Edit web ACL**.
   d. Choose the **x** at the right of the rule heading.
   e. Choose **Actions**, and then choose **Delete web ACL**.
4. Delete your rule:

   a. In the navigation pane, choose **Rules**.

   b. Choose the rule that you created during the tutorial.
   c. Choose **Delete**.
   d. In the **Delete** dialog box, choose **Delete** again to confirm.

AWS WAF doesn't charge for conditions, but if you want to complete the cleanup, perform the following procedure to remove filters from conditions and delete the conditions.

#### To delete filters and conditions

1. Delete the IP address range in your IP match condition, and delete the IP match condition:

   a. In the navigation pane of the AWS WAF console, choose **IP addresses**.
   b. Choose the IP match condition that you created during the tutorial.
   c. Select the check box for the IP address range that you added.
   d. Choose **Delete IP address or range**.
   e. In the **IP match conditions** pane, choose **Delete**.
   f. In the **Delete** dialog box, choose **Delete** again to confirm.
2. Delete the filter in your SQL injection match condition, and delete the SQL injection match condition:

   a. In the navigation pane, choose **SQL injection**.
   b. Choose the SQL injection match condition that you created during the tutorial.
   c. Select the check box for the filter that you added.
   d. Choose **Delete filter**.
   e. In the **SQL injection match conditions** pane, choose **Delete**.
   f. In the **Delete** dialog box, choose **Delete** again to confirm.
3. Delete the filter in your string match condition, and delete the string match condition:

   a. In the navigation pane, choose **String matching**.
   b. Choose the string match condition that you created during the tutorial.
   c. Select the check box for the filter that you added.
   d. Choose **Delete filter**.
   e. In the **String match conditions** pane, choose **Delete**.
   f. In the **Delete** dialog box, choose **Delete** again to confirm.
4. Delete the filter in your size constraint condition, and delete the size constraint condition:

   a. In the navigation pane, choose **Size constraints**.
   b. Choose the size constraint condition that you created during the tutorial.
   c. Select the check box for the filter that you added.
   d. Choose **Delete filter**.
   e. In the **Size constraint conditions** pane, choose **Delete**.
   f. In the **Delete** dialog box, choose **Delete** again to confirm.

# Tutorials

This section contains a link to a preconfigured template as well as three tutorials that present complete solutions for common tasks that you can perform in AWS WAF. The tutorials show how to combine several AWS services to automatically configure AWS WAF in response to your CloudFront traffic.

Their purpose is to provide general guidance. They are not intended for direct use in your production environment without careful review and adaptation to the unique aspects of your business environment.

**AWS WAF Preconfigured Protections**

You can use our preconfigured template to get started quickly with AWS WAF. The template includes a set of AWS WAF rules that are designed to block common web-based attacks. You can customize the template to fit your business needs.

The rules in the template help protect against bad bots, SQL injection, cross-site scripting (XSS), HTTP floods, and other known attacks. After you deploy the template, AWS WAF begins to block the web requests to your CloudFront distributions or Application Load Balancers that match the preconfigured rules in your web access control (web ACL) list. You can use this automated solution in addition to other web ACLs that you configure. For more information, see AWS WAF Security Automations.

**Tutorials**

- Tutorial: Quickly Setting Up AWS WAF Protection Against Common Attacks (p. 18)
- Tutorial: Blocking IP Addresses That Submit Bad Requests (p. 24)

# Tutorial: Quickly Setting Up AWS WAF Protection Against Common Attacks

This tutorial shows you how to use AWS CloudFormation to quickly configure AWS WAF to protect against the following common attacks:

- **Cross-site scripting attacks** – Attackers sometimes insert scripts into web requests in an effort to exploit vulnerabilities in web applications. Cross-site scripting match conditions identify the parts of web requests, such as the URI or the query string, that you want AWS WAF to inspect for possible malicious scripts.
- **SQL injection attacks** – Attackers sometimes insert malicious SQL code into web requests in an effort to extract data from your database. SQL injection match conditions identify the part of web requests that you want AWS WAF to inspect for possible malicious SQL code.
- **Attacks from known bad IP addresses** – You can use IP match conditions to allow, block, or count web requests based on the IP addresses that the requests originate from. An IP match condition lists up to 1,000 IP addresses or IP address ranges that you specify.

> **Note**
> This tutorial assumes that you have a CloudFront distribution that you use to deliver content for your web application. If you don't have a CloudFront distribution, see Creating or Updating a Web Distribution Using the CloudFront Console in the *Amazon CloudFront Developer Guide*.

Topics

## Solution Overview

AWS CloudFormation uses a template to set up the following AWS WAF conditions, rules, and a web ACL.

### Conditions

AWS CloudFormation creates the following conditions.

**IP Match Condition**

Filters requests that come from known bad IP addresses. This lets you easily add IPs to a list to block access to your website. You might want to do this if you're receiving a lot of bad requests from one or more IP addresses. If you want to allow, block, or count requests based on the IP addresses that the requests come from, see Step 3: (Optional) Add IP Addresses to the IP Match Condition (p. 22) later in this tutorial.

The name of the condition is *prefix***ManualBlockSet** where *prefix* is the name that you specify for the web ACL when you create the AWS CloudFormation stack.

**Size Constraint Condition**

Filters requests for which the body is longer than 8,192 bytes. AWS WAF evaluates only the first 8,192 bytes of the request part that you specify in a filter. If valid request bodies never exceed 8,192 bytes, you can use a size constraint condition to catch malicious requests that might otherwise slip through.

For this tutorial, AWS CloudFormation configures AWS WAF only to count, not block, requests that have a body longer than 8,192 bytes. If the body in your requests never exceeds that length, you can change the configuration to block requests that have longer bodies. For information about how to view the count of requests that exceed 8,192 bytes and how to change the web ACL to block requests that contain bodies larger than 8,192 bytes, see Step 4: (Optional) Update the Web ACL to Block Large Bodies (p. 23).

The name of the condition is *prefix***LargeBodyMatch** where *prefix* is the name that you specify for the web ACL when you create the AWS CloudFormation stack.

**SQL Injection Condition**

Filters requests that contain possible malicious SQL code. The condition includes filters that evaluate the following parts of requests:

- Query string (URL decode transformation)
- URI (URL decode transformation)
- Body (URL decode transformation)
- Body (HTML decode transformation)

The name of the condition is *prefix***SqliMatch** where *prefix* is the name that you specify for the web ACL when you create the AWS CloudFormation stack.

**Cross-site Scripting Condition**

Filters requests that contain possible malicious scripts. The condition includes filters that evaluate the following parts of requests:

- Query string (URL decode transformation)
- URI (URL decode transformation)
- Body (URL decode transformation)
- Body (HTML decode transformation)

The name of the condition is *prefix***XssMatch** where *prefix* is the name that you specify for the web ACL when you create the AWS CloudFormation stack.

## Rules

When you create the AWS CloudFormation stack, AWS CloudFormation creates the following rules and adds the corresponding condition to each rule:

***prefix*ManualIPBlockRule**

AWS CloudFormation adds the *prefix***ManualBlockSet** condition to this rule.

***prefix*SizeMatchRule**

AWS CloudFormation adds the *prefix***LargeBodyMatch** condition to this rule.

***prefix*SqliRule**

AWS CloudFormation adds the *prefix***SqliMatch** condition to this rule.

***prefix*XssRule**

AWS CloudFormation adds the *prefix***XssMatch** condition to this rule.

## Web ACL

AWS CloudFormation creates a web ACL that has the name that you specify when you create the AWS CloudFormation stack. The web ACL contains the following rules with the specified settings:

***prefix*ManualIPBlockRule**

By default, the condition in this rule doesn't contain any IP addresses. If you want to allow, block, or count requests based on the IP addresses that the requests come from, see Step 3: (Optional) Add IP Addresses to the IP Match Condition (p. 22) later in this tutorial.

***prefix*SizeMatchRule**

By default, AWS WAF counts requests for which the body is longer than 8,192 bytes.

***prefix*SqliRule**

AWS WAF blocks requests based on the settings in this rule.

***prefix*XssRule**

AWS WAF blocks requests based on the settings in this rule.

## Requirements

This tutorial assumes that you have a CloudFront distribution that you use to deliver content for your web application. If you don't have a CloudFront distribution, see Creating or Updating a Web Distribution Using the CloudFront Console in the *Amazon CloudFront Developer Guide*. This tutorial also uses AWS CloudFormation to simplify the provisioning process. For more information, see the AWS CloudFormation User Guide.

## Estimated Time

The estimated time to complete this tutorial is 15 minutes if you already have a CloudFront distribution, or 30 minutes if you need to create a CloudFront distribution.

## Costs

There is a cost associated with the resources that you create during this tutorial. You can delete the resources after you finish the tutorial to stop incurring charges. For more information, see AWS WAF Pricing and Amazon CloudFront Pricing.

## Step 1: Create an AWS CloudFormation Stack That Sets Up AWS WAF Protection Against Common Attacks

In the following procedure, you use an AWS CloudFormation template to create a stack that sets up AWS WAF protection against common attacks.

> **Important**
> You begin to incur charges for the different services when you create the AWS CloudFormation stack that deploys this solution. Charges continue to accrue until you delete the AWS CloudFormation stack. For more information, see Step 5: (Optional) Delete Your AWS CloudFormation Stack (p. 23).

**To create an AWS CloudFormation stack for blocking IP addresses that submit bad requests**

1. To start the process that creates an AWS CloudFormation stack, choose the link for the region in which you want to create AWS resources:
   - Create a stack in US East (N. Virginia)
   - Create a stack in US West (Oregon)
   - Create a stack in EU (Ireland)
   - Create a stack in Asia Pacific (Tokyo)
2. If you are not already signed in to the AWS Management Console, sign in when prompted.
3. On the **Select Template** page, choose **Specify an Amazon S3 template URL**. For the template URL, type `https://s3.amazonaws.com/cloudformation-examples/community/common-attacks.json`.
4. Choose **Next**.
5. On the **Specify Details** page, specify the following values:

   **Stack Name**

   You can use the default name (**CommonAttackProtection**), or you can change the name. The stack name must not contain spaces and must be unique within your AWS account.

   **Name**

   Specify a name for the web ACL that AWS CloudFormation will create. The name that you specify is also used as a prefix for the conditions and rules that AWS CloudFormation will create, so you can easily find all the related objects.
6. Choose **Next**.
7. (Optional) On the **Options** page, enter tags and advanced settings or leave the boxes blank.
8. Choose **Next**.
9. On the **Review** page, review the configuration, and then choose **Create**.

   After you choose **Create**, AWS CloudFormation creates the AWS WAF resources that are identified in Solution Overview (p. 19).

## Step 2: Associate a Web ACL with a CloudFront Distribution

After AWS CloudFormation creates the stack, you must associate your CloudFront distribution to activate AWS WAF and update your Amazon S3 bucket to enable event notification.

> **Note**
> You can associate a web ACL with as many distributions as you want, but you can associate only one web ACL with a given distribution.

### To associate a web ACL with a CloudFront distribution

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Web ACLs**.
3. Choose the web ACL that you want to associate with a CloudFront distribution.
4. On the **Rules** tab, under **AWS resources using this web ACL**, choose **Add association**.
5. When prompted, use the **Resource** list to choose the distribution that you want to associate this web ACL with.
6. Choose **Add**.
7. To associate this web ACL with additional CloudFront distributions, repeat steps 4 through 6.

If you already have an Amazon S3 bucket for CloudFront access logs (if you selected **no** for **Create CloudFront Access Log Bucket** in the preceding procedure), enable Amazon S3 event notification to trigger the Lambda function when a new log file is added to the bucket. For more information, see Enabling Event Notifications in the *Amazon Simple Storage Service Console User Guide*.

> **Note**
> If you chose to have AWS CloudFormation create the bucket for you, AWS CloudFormation also enabled event notifications for the bucket.

### To enable Amazon S3 event notification

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose the bucket that you want to use for CloudFront access logs.
3. Choose **Properties**, and expand **Events**.
4. Specify the following values:

   **Name**

   Type a name for the event, such as **LambdaNotificationsForWAFBadRequests**. The name cannot contain spaces.

   **Events**

   Select **ObjectCreated (All)**.

   **Prefix**

   Leave the field empty.

   **Suffix**

   Type **gz**.

   **Send To**

   Select **Lambda function**.

   **Lambda function**

   Choose **BadBehavingIP** or the name that you specified for your AWS CloudFormation stack.
5. Choose **Save**.

## Step 3: (Optional) Add IP Addresses to the IP Match Condition

When you created the AWS CloudFormation stack, AWS CloudFormation created an IP match condition for you, added it to a rule, added the rule to a web ACL, and configured the web ACL to block requests based on IP addresses. The IP match condition doesn't include any IP addresses, though. If you want to block requests based on IP addresses, perform the following procedure.

### To edit AWS CloudFormation parameter values

1. Open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **IP addresses**.
3. In the **IP match conditions** pane, choose the IP match condition that you want to edit.
4. To add an IP address range:

   a. In the right pane, choose **Add IP address or range**.

   b. Type an IP address or range by using CIDR notation. Here are two examples:

      - To specify the IP address 192.0.2.44, type **192.0.2.44/32**.
      - To specify the range of IP addresses from 192.0.2.0 to 192.0.2.255, type **192.0.2.0/24**.

      AWS WAF supports /8, /16, /24, and /32 IP address ranges. For more information about CIDR notation, see the Wikipedia entry Classless Inter-Domain Routing.

      > **Note**
      > AWS WAF supports both IPv4 and IPv6 IP addresses.

   c. To add more IP addresses, choose **Add another IP address**, and then type the value.

   d. Choose **Add**.

## Step 4: (Optional) Update the Web ACL to Block Large Bodies

When you created the AWS CloudFormation stack, AWS CloudFormation created a size constraint condition that filters requests that have request bodies longer than 8,192 bytes. It also added the condition to a rule, and added the rule to the web ACL. In this example, AWS CloudFormation configured the web ACL to count requests, not to block requests. This is useful when you want to confirm you are not blocking valid requests inadvertently.

If you want to block requests that are longer than 8,192 bytes, perform the following procedure.

### To change the action for a rule in a web ACL

1. Open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Web ACLs**.
3. Choose the web ACL that you want to edit.
4. In the right pane, choose the **Rules** tab.
5. Choose **Edit Web ACL**.
6. To change the action for the *prefix***LargeBodyMatchRule**, choose the preferred option. (*prefix* is the value that you specified for the name of the web ACL.)
7. Choose **Save changes**.

## Step 5: (Optional) Delete Your AWS CloudFormation Stack

If you want to stop protecting from common attacks as described in Solution Overview (p. 19), delete the AWS CloudFormation stack that you created in Step 1: Create an AWS CloudFormation Stack That Sets Up AWS WAF Protection Against Common Attacks (p. 21). This deletes the AWS WAF resources that AWS CloudFormation created and stops the AWS charges for those resources.

**To delete an AWS CloudFormation stack**

1. Sign in to the AWS Management Console and open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.
2. Select the check box for the stack. The default name is **CommonAttackProtection**.
3. Choose **Delete Stack**.
4. Choose **Yes, Delete** to confirm.
5. To track the progress of the stack deletion, select the check box for the stack, and choose the **Events** tab in the bottom pane.

## Related Resources

For AWS WAF samples, including Lambda functions, AWS CloudFormation templates, and SDK usage examples, go to GitHub at https://github.com/awslabs/aws-waf-sample.

# Tutorial: Blocking IP Addresses That Submit Bad Requests

Using AWS Lambda, you can set a threshold of how many bad requests per minute your web application will tolerate from a given IP address. A bad request is one for which your CloudFront origin returns one of the following HTTP 40x status codes:

- 400, Bad Request
- 403, Forbidden
- 404, Not Found
- 405, Method Not Allowed

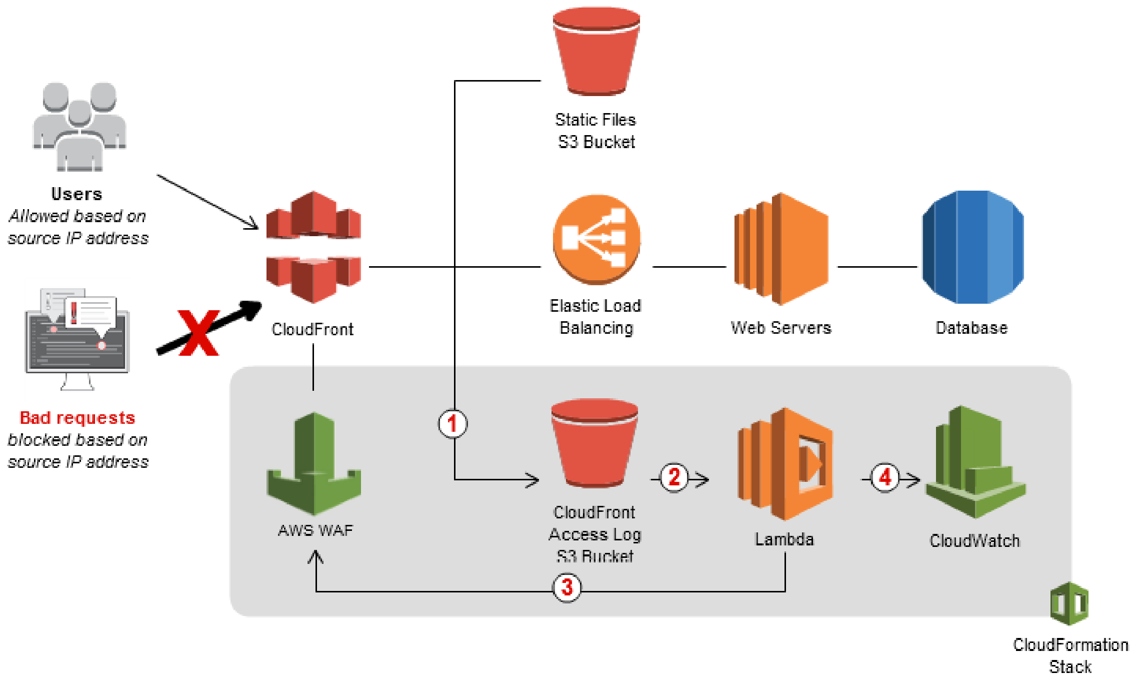
If users (based on IP addresses) exceed this error code threshold, Lambda automatically updates your AWS WAF rules to block IP addresses and specify for how long requests from those IP addresses should be blocked.

This tutorial shows you how to use an AWS CloudFormation template to specify the request threshold and time to block requests. The tutorial also uses CloudFront access logs (stored in Amazon S3) to count requests as they are served by CloudFront and by Amazon CloudWatch metrics.

Topics

## Solution Overview

The following illustration shows how you can use AWS WAF with AWS Lambda to block requests from specific IP addresses.



1. As CloudFront receives requests on behalf of your web application, it sends access logs to an Amazon S3 bucket that contains detailed information about the requests.
2. For every new access log stored in the Amazon S3 bucket, a Lambda function is triggered. The Lambda function parses the log files and looks for requests that resulted in error codes 400, 403, 404, and 405. The function then counts the number of bad requests and temporarily stores results in `current_outstanding_requesters.json` in the Amazon S3 bucket that you're using for access logs.
3. The Lambda function updates AWS WAF rules to block the IP addresses that are listed in `current_outstanding_requesters.json` for a period of time that you specify. After this blocking period has expired, AWS WAF allows those IP addresses to access your application again, but continues to monitor the requests from those IP addresses.
4. The Lambda function publishes execution metrics in CloudWatch, such as the number of requests analyzed and IP addresses blocked.

The AWS CloudFormation template creates a web access control list (web ACL) and two separate rules in AWS WAF that block and monitor requests from IP addresses, depending on the settings that you configure during the tutorial. The two rules are defined here:

- **Auto Block** – This rule adds IP addresses that exceed the request-per-minute limit. New requests from those IP addresses are blocked until Lambda removes the IP addresses from the block list after the specified expiration period. The default is four hours.
- **Manual Block** – This rule adds IP addresses manually to the auto-block list. The IP addresses are permanently blocked; they can access the web application only if you remove them from the block list. You can use this list to block known bad IP addresses or IP addresses that frequently are added to the auto-block rule.

**Requirements:** This tutorial assumes that you already have a CloudFront distribution that you use to deliver content for your web application. If you don't have a CloudFront distribution, see Creating or Updating a Web Distribution Using the CloudFront Console in the *Amazon CloudFront Developer Guide*. This tutorial also uses AWS CloudFormation to simplify the provisioning process. For more information, see the AWS CloudFormation User Guide.

**Estimated time:** 15 minutes if you already have a CloudFront distribution, or 30 minutes if you need to create a CloudFront distribution.

**Estimated cost:**

- **AWS WAF**
  - $5.00 per month per web ACL (the tutorial creates one web ACL)
  - $1.00 per month per rule (x2 for the two rules that AWS CloudFormation creates for this tutorial)
  - $0.60 per million requests
- **AWS Lambda** – Each new CloudFront access log represents a new request and triggers the Lambda function that is created by this tutorial. Lambda charges include the following:
  - **Requests** – The first million requests are free, and then Lambda charges $0.20 per million requests. CloudFront delivers access logs for a distribution up to several times an hour.
  - **Memory used per second** – $0.00001667 per GB of memory used per second.
- **Amazon S3** – Amazon S3 charges for storing CloudFront access logs. The size of the logs and, therefore, the charge for storage depends on the number of requests that CloudFront receives for your objects. For more information, see Amazon S3 Pricing.
- **CloudFront** – You don't incur any additional CloudFront charges for this solution. For more information, see Amazon CloudFront Pricing.

## Step 1: Create an AWS CloudFormation Stack for Blocking IP Addresses That Submit Bad Requests

In the following procedure, you use an AWS CloudFormation template to create a stack that launches the AWS resources required by Lambda, CloudFront, Amazon S3, AWS WAF, and CloudWatch.

> **Important**
> You begin to incur charges for the different services when you create the AWS CloudFormation stack that deploys this solution. Charges continue to accrue until you delete the AWS CloudFormation stack. For more information, see Step 5: (Optional) Delete Your AWS CloudFormation Stack (p. 29).

**To create an AWS CloudFormation stack for blocking IP addresses that submit bad requests**

1. To start the process that creates an AWS CloudFormation stack, choose the link for the region in which you want to create AWS resources:
   - Create a stack in US East (N. Virginia)
   - Create a stack in US West (Oregon)
   - Create a stack in EU (Ireland)
   - Create a stack in Asia Pacific (Tokyo)
2. If you are not already signed in to the AWS Management Console, sign in when prompted.
3. On the **Select Template** page, the selected URL automatically appears under **Specify an Amazon S3 template URL**. Choose **Next**.
4. On the **Specify Details** page, specify the following values:

   **Stack Name**

   You can use the default name (**BadBehavingIP**), or you can change the name. The stack name must not contain spaces and must be unique within your AWS account.

   **Create CloudFront Access Log Bucket**

   Select **yes** to create a new Amazon S3 bucket for CloudFront access logs, or select **no** if you already have an Amazon S3 bucket for CloudFront access logs.

**CloudFront Access Log Bucket Name**

Type the name of the Amazon S3 bucket where you want CloudFront to put access logs. Leave this box empty if you selected **no** for **Create CloudFront Access Log Bucket**.

**Request Threshold**

Type the maximum number of requests that can be made from an IP address per minute without being blocked. The default is 400.

**WAF Block Period**

Specify how long (in minutes) an IP address should be blocked after crossing the threshold. The default is 240 minutes (four hours).

5. Choose **Next**.
6. (Optional) On the **Options** page, enter tags and advanced settings or leave the boxes blank.
7. Choose **Next**.
8. On the **Review** page, select the **I acknowledge** check box, and then choose **Create**.

   After you choose **Create**, AWS CloudFormation creates the AWS resources that are necessary to run the solution:

   - Lambda function
   - AWS WAF web ACL (named **Malicious Requesters**) with the necessary rules configured
   - CloudWatch custom metric
   - Amazon S3 bucket with the name that you specified in the **CloudFront Access Log Bucket Name** field in step 6, if you selected **yes** for **Create CloudFront Access Log Bucket**

## Step 2: Associate a Web ACL with a CloudFront Distribution

After AWS CloudFormation creates the stack, you must associate the CloudFront distribution to activate AWS WAF and update your Amazon S3 bucket to enable event notification.

> **Note**
> If you're already using AWS WAF to monitor CloudFront requests and if logging is already enabled for the distribution that you're monitoring, you can skip the first procedure.

> **Note**
> You can associate a web ACL with as many distributions as you want, but you can associate only one web ACL with a given distribution.

### To associate a web ACL with a CloudFront distribution

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Web ACLs**.
3. Choose the web ACL that you want to associate with a CloudFront distribution.
4. On the **Rules** tab, under **AWS resources using this web ACL**, choose **Add association**.
5. When prompted, use the **Resource** list to choose the distribution that you want to associate this web ACL with.
6. Choose **Add**.
7. To associate this web ACL with additional CloudFront distributions, repeat steps 4 through 6.

If you already have an Amazon S3 bucket for CloudFront access logs (if you selected **no** for **Create CloudFront Access Log Bucket** in the preceding procedure), enable Amazon S3 event notification to trigger the Lambda function when a new log file is added to the bucket. For more information, see Enabling Event Notifications in the *Amazon Simple Storage Service Console User Guide*.

> **Note**
> If you chose to have AWS CloudFormation create the bucket for you, AWS CloudFormation also enabled event notifications for the bucket.

### To enable Amazon S3 event notification

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose the bucket that you want to use for CloudFront access logs.
3. Choose **Properties**, and expand **Events**.
4. Specify the following values:

   **Name**

   Type a name for the event, such as **LambdaNotificationsForWAFBadRequests**. The name cannot contain spaces.

   **Events**

   Select **ObjectCreated (All)**.

   **Prefix**

   Leave the field empty.

   **Suffix**

   Type **gz**.

   **Send To**

   Select **Lambda function**.

   **Lambda function**

   Choose **BadBehavingIP** or the name that you specified for your AWS CloudFormation stack.
5. Choose **Save**.

## Step 3: (Optional) Edit AWS CloudFormation Parameter Values

If you want to change the parameters after you create the AWS CloudFormation stack—for example, if you want to change the threshold value or how long IPs are blocked—you can update the AWS CloudFormation stack.

### To edit AWS CloudFormation parameter values

1. Open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.
2. In the list of stacks, choose the running stack that you want to update, which is **BadBehavingIP** if you accepted the default value when you created the stack.
3. Choose **Actions**, and then choose **Update Stack**.
4. On the **Select Template** page, select **Use current template**, and then choose **Next**.
5. On the **Specify Details** page, change the values of **Error Code Blacklisting Parameters** as applicable:

   **Request Threshold**

   Type the new maximum number of requests that can be made per minute without being blocked.

**WAF Block Period**

Specify the new value of how long (in minutes) that you want AWS WAF to block the IP address after the number of requests from that IP address exceed the value of **Request Threshold**.

6. On the **Options** page, choose **Next**.
7. On the **Review** page, select the **I acknowledge** check box, and then choose **Update**.

   AWS CloudFormation updates the stack to reflect the new values of the parameters.

## Step 4: (Optional) Test Your Thresholds and IP Rules

To test your solution, you can wait until CloudFront generates a new access log file, or you can simulate this process by uploading a sample access log into the Amazon S3 bucket that you specified for receiving log files.

### To test your thresholds and IP rules

1. Download the sample CloudFront access log file from the AWS website.
2. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
3. Choose the Amazon S3 bucket that you're using for CloudFront access logs for this tutorial.
4. Choose **Upload**.
5. Choose **Add Files**, choose the sample access log file, and then choose **Start Upload**.

After the upload finishes, perform the following procedure to confirm that the IP addresses were populated automatically in the AWS WAF **Auto Block** rule. Lambda takes a few seconds to process the log file and update the rule.

### To review IP addresses in the Auto Block rule

1. Open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Rules**.
3. Choose the **Auto Block** rule.
4. Confirm that the **Auto Block** rule includes an IP match condition that contains IP addresses.

## Step 5: (Optional) Delete Your AWS CloudFormation Stack

If you want to stop blocking IP addresses that submit bad requests, delete the AWS CloudFormation stack that you created in Step 1: Create an AWS CloudFormation Stack for Blocking IP Addresses That Submit Bad Requests (p. 26). This deletes the AWS resources that AWS CloudFormation created and stops the AWS charges for those resources.

### To delete an AWS CloudFormation stack

1. Sign in to the AWS Management Console and open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/.
2. Select the check box for the stack. The default name is **BadBehavingIP**.
3. Choose **Delete Stack**.
4. Choose **Yes, Delete** to confirm.
5. To track the progress of the stack deletion, select the check box for the stack, and then choose the **Events** tab in the bottom pane.

## Related Resources

For AWS WAF samples, including Lambda functions, AWS CloudFormation templates, and SDK usage examples, go to GitHub at https://github.com/awslabs/aws-waf-sample.

**Blog Tutorials**

The following tutorial topics link out to the AWS Security Blog.

- How to Import IP Address Reputation Lists to Automatically Update AWS WAF IP Blacklists
- How to Reduce Security Threats and Operating Costs Using AWS WAF and Amazon CloudFront
- How to Prevent Hotlinking by Using AWS WAF, Amazon CloudFront, and Referer Checking
- How to Use AWS CloudFormation to Automate Your AWS WAF Configuration with Example Rules and Match Conditions

# Creating and Configuring a Web Access Control List (Web ACL)

A web access control list (web ACL) gives you fine-grained control over the web requests that your Amazon CloudFront distributions or Application Load Balancers, respond to. You can allow or block the following types of requests:

- Originate from an IP address or a range of IP addresses
- Contain a specified string in a particular part of requests
- Exceed a specified length
- Appear to contain malicious SQL code (known as SQL injection)
- Appear to contain malicious scripts (known as cross-site scripting)

You can also test for any combination of these conditions, or block or count web requests that not only meet the specified conditions, but also exceed a specified number of requests in any 5-minute period.

To choose the requests that you want to allow to have access to your content or that you want to block, perform the following tasks:

1. Choose the default action, allow or block, for web requests that don't match any of the conditions that you specify. For more information, see Deciding on the Default Action for a Web ACL (p. 31).
2. Specify the conditions under which you want to allow or block requests:
   - To allow or block requests based on whether the requests appear to contain malicious scripts, create cross-site scripting match conditions. For more information, see Working with Cross-site Scripting Match Conditions (p. 31).
   - To allow or block requests based on the IP addresses that they originate from, create IP match conditions. For more information, see Working with IP Match Conditions (p. 35).
   - To allow or block requests based on whether the requests exceed a specified length, create size constraint conditions. For more information, see Working with Size Constraint Conditions (p. 37).
   - To allow or block requests based on whether the requests appear to contain malicious SQL code, create SQL injection match conditions. For more information, see Working with SQL Injection Match Conditions (p. 42).
   - To allow or block requests based on strings that appear in the requests, create string match conditions. For more information, see Working with String Match Conditions (p. 45).

3. Add the conditions to one or more rules. If you add more than one condition to the same rule, web requests must match all the conditions for AWS WAF to allow or block requests based on the rule. For more information, see Working with Rules (p. 50). Optionally, also at a rate limit to the rule, which specifies the maximum number of requests allowed from a specific IP address.
4. Add the rules to a web ACL. For each rule, specify whether you want AWS WAF to allow or block requests based on the conditions that you added to the rule. If you add more than one rule to a web ACL, AWS WAF evaluates the rules in the order that they're listed in the web ACL. For more information, see Working with Web ACLs (p. 53).

Topics

# Deciding on the Default Action for a Web ACL

When you create and configure a web ACL, the first and most important decision that you must make is whether the default action should be for AWS WAF to allow web requests or to block web requests. The default action indicates what you want AWS WAF to do after it inspects a web request for all the conditions that you specify, and the web request doesn't match any of those conditions:

- **Allow** – If you want to allow most users to access your website, but you want to block access to attackers whose requests originate from specified IP addresses, or whose requests appear to contain malicious SQL code or specified values, choose **Allow** for the default action.
- **Block** – If you want to prevent most would-be users from accessing your website, but you want to allow access to users whose requests originate from specified IP addresses, or whose requests contain specified values, choose **Block** for the default action.

Many decisions that you make after you've decided on a default action depend on whether you want to allow or block most web requests. For example, if you want to *allow* most requests, then the match conditions that you create generally should specify the web requests that you want to *block*, such as the following:

- Requests that originate from IP addresses that are making an unreasonable number of requests
- Requests that include fake values in the **User-Agent** header
- Requests that appear to include malicious SQL code

# Working with Cross-site Scripting Match Conditions

Attackers sometimes insert scripts into web requests in an effort to exploit vulnerabilities in web applications. You can create one or more cross-site scripting match conditions to identify the parts of web requests, such as the URI or the query string, that you want AWS WAF to inspect for possible malicious scripts. Later in the process, when you create a web ACL, you specify whether to allow or block requests that appear to contain malicious scripts.

Topics

## Creating Cross-site Scripting Match Conditions

When you create cross-site scripting match conditions, you specify filters, which indicate the part of web requests that you want AWS WAF to inspect for malicious scripts, such as the URI or the query string. You can add more than one filter to a cross-site scripting match condition, or you can create a separate condition for each filter. Here's how each configuration affects AWS WAF behavior:

- **More than one filter per cross-site scripting match condition (recommended)** – When you add a cross-site scripting match condition that contains multiple filters to a rule and add the rule to a web ACL, a web request must match only one of the filters in the cross-site scripting match condition for AWS WAF to allow or block the request based on that condition.

  For example, suppose you create one cross-site scripting match condition, and the condition contains two filters. One filter instructs AWS WAF to inspect the URI for malicious scripts, and the other instructs AWS WAF to inspect the query string. AWS WAF allows or blocks requests if they appear to contain malicious scripts *either* in the URI *or* in the query string.
- **One filter per cross-site scripting match condition** – When you add the separate cross-site scripting match conditions to a rule and add the rule to a web ACL, web requests must match all the conditions for AWS WAF to allow or block requests based on the conditions.

  Suppose you create two conditions, and each condition contains one of the two filters in the preceding example. When you add both conditions to the same rule and add the rule to a web ACL, AWS WAF allows or blocks requests only when both the URI and the query string appear to contain malicious scripts.

> **Note**
> When you add a cross-site scripting match condition to a rule, you also can configure AWS WAF to allow or block web requests that *do not* appear to contain malicious scripts.

**To create a cross-site scripting match condition**

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Cross-site scripting**.
3. Choose **Create condition**.
4. Specify the applicable filter settings. For more information, see Values That You Specify When You Create or Edit Cross-site Scripting Match Conditions (p. 32).
5. Choose **Add another filter**.
6. If you want to add another filter, repeat steps 4 and 5.
7. When you're done adding filters, choose **Create**.

## Values That You Specify When You Create or Edit Cross-site Scripting Match Conditions

When you create or update a cross-site scripting match condition, you specify the following values:

**Name**

The name of the cross-site scripting match condition.

The name can contain only the characters A-Z, a-z, and 0-9. You can't change the name of a condition after you create it.

**Part of the request to filter on**

Choose the part of each web request that you want AWS WAF to inspect for malicious scripts:

**Header**

A specified request header, for example, the `User-Agent` or `Referer` header. If you choose **Header**, specify the name of the header in the **Header** field.

**HTTP method**

The HTTP method, which indicates the type of operation that the request is asking the origin to perform. CloudFront supports the following methods: `DELETE`, `GET`, `HEAD`, `OPTIONS`, `PATCH`, `POST`, and `PUT`.

**Query string**

The part of a URL that appears after a `?` character, if any.

**URI**

The part of a URL that identifies a resource, for example, `/images/daily-ad.jpg`.

**Body**

The part of a request that contains any additional data that you want to send to your web server as the HTTP request body, such as data from a form.

> **Note**
> If you choose **Body** for the value of **Part of the request to filter on**, AWS WAF inspects only the first 8192 bytes (8 KB). To allow or block requests for which the body is longer than 8192 bytes, you can create a size constraint condition. (AWS WAF gets the length of the body from the request headers.) For more information, see Working with Size Constraint Conditions (p. 37).

**Header**

If you chose **Header** for **Part of the request to filter on**, choose a header from the list of common headers, or type the name of a header that you want AWS WAF to inspect for malicious scripts.

**Transformation**

A transformation reformats a web request before AWS WAF inspects the request. This eliminates some of the unusual formatting that attackers use in web requests in an effort to bypass AWS WAF. Transformations can perform the following operations:

**None**

AWS WAF doesn't perform any text transformations on the web request before inspecting it for the string in **Value to match**.

**Convert to lowercase**

AWS WAF converts uppercase letters (A-Z) to lowercase (a-z).

**HTML decode**

AWS WAF replaces HTML-encoded characters with unencoded characters:

- Replaces `&quot;` with `&`
- Replaces `&nbsp;` with a non-breaking space
- Replaces `&lt;` with `<`
- Replaces `&gt;` with `>`
- Replaces characters that are represented in hexadecimal format, `&#xhhhh;`, with the corresponding characters
- Replaces characters that are represented in decimal format, `&#nnnn;`, with the corresponding characters

**Normalize whitespace**

AWS WAF replaces the following characters with a space character (decimal 32):

- \f, formfeed, decimal 12
- \t, tab, decimal 9
- \n, newline, decimal 10
- \r, carriage return, decimal 13
- \v, vertical tab, decimal 11
- non-breaking space, decimal 160

In addition, this option replaces multiple spaces with one space.

**Simplify command line**

For requests that contain operating system command line commands, use this option to perform the following transformations:

- Delete the following characters: \ " ' ^
- Delete spaces before the following characters: / (
- Replace the following characters with a space: , ;
- Replace multiple spaces with one space
- Convert uppercase letters (A-Z) to lowercase (a-z)

**URL decode**

Decode a URL-encoded request.

## Adding and Deleting Filters in a Cross-site Scripting Match Condition

You can add or delete filters in a cross-site scripting match condition. To change a filter, add a new one and delete the old one.

### To add or delete filters in a cross-site scripting match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Cross-site scripting**.
3. Choose the condition that you want to add or delete filters in.
4. To add filters, perform the following steps:

   a. Choose **Add filter**.

   b. Specify the applicable filter settings. For more information, see Values That You Specify When You Create or Edit Cross-site Scripting Match Conditions (p. 32).
   c. Choose **Add**.
5. To delete filters, perform the following steps:

   a. Select the filter that you want to delete.
   b. Choose **Delete filter**.

## Deleting Cross-site Scripting Match Conditions

If you want to delete a cross-site scripting match condition, you must first delete all filters in the condition and remove the condition from all the rules that are using it, as described in the following procedure.

**To delete a cross-site scripting match condition**

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Cross-site scripting**.
3. In the **Cross-site scripting match conditions** pane, choose the cross-site scripting match condition that you want to delete.
4. In the right pane, choose the **Associated rules** tab.

   If the list of rules using this cross-site scripting match condition is empty, go to step 6. If the list contains any rules, make note of the rules, and continue with step 5.
5. To remove the cross-site scripting match condition from the rules that are using it, perform the following steps:

   a. In the navigation pane, choose **Rules**.
   b. Choose the name of a rule that is using the cross-site scripting match condition that you want to delete.
   c. In the right pane, select the cross-site scripting match condition that you want to remove from the rule, and choose **Remove selected condition**.
   d. Repeat steps b and c for all the remaining rules that are using the cross-site scripting match condition that you want to delete.
   e. In the navigation pane, choose **Cross-site scripting**.
   f. In the **Cross-site scripting match conditions** pane, choose the cross-site scripting match condition that you want to delete.
6. Choose **Delete** to delete the selected condition.

# Working with IP Match Conditions

If you want to allow or block web requests based on the IP addresses that the requests originate from, create one or more IP match conditions. An IP match condition lists up to 10,000 IP addresses or IP address ranges that your requests originate from. Later in the process, when you create a web ACL, you specify whether to allow or block requests from those IP addresses.

Topics

## Creating an IP Match Condition

If you want to allow some web requests and block others based on the IP addresses that the requests originate from, create an IP match condition for the IP addresses that you want to allow and another IP match condition for the IP addresses that you want to block.

> **Note**
> When you add an IP match condition to a rule, you also can configure AWS WAF to allow or block web requests that *do not* originate from the IP addresses that you specify in the condition.

### To create an IP match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **IP addresses**.
3. Choose **Create condition**.
4. Type a name in the **Name** field.

   The name can contain only the characters A-Z, a-z, and 0-9. You can't change the name of a condition after you create it.
5. Select the correct IP version and specify an IP address or range of IP addresses by using CIDR notation. Here are some examples:

   - To specify the IPv4 address 192.0.2.44, type **192.0.2.44/32**.
   - To specify the IPv6 address 0:0:0:0:0:ffff:c000:22c, type **0:0:0:0:0:ffff:c000:22c/128**.
   - To specify the range of IPv4 addresses from 192.0.2.0 to 192.0.2.255, type **192.0.2.0/24**.
   - To specify the range of IPv6 addresses from 2620:0:2d0:200:0:0:0:0 to 2620:0:2d0:200:ffff:ffff:ffff:ffff, type **2620:0:2d0:200::/64**.

   AWS WAF supports /8, /16, /24, and /32 IPv4 address ranges and /16, /24, /32, /56, /64, and /128 IPv6 address ranges. For more information about CIDR notation, see the Wikipedia entry Classless Inter-Domain Routing.
6. Choose **Add another IP address or range**.
7. If you want to add another IP address or range, repeat steps 5 and 6.
8. When you're finished adding values, choose **Create IP match condition**.

## Editing IP Match Conditions

You can add an IP address range to an IP match condition or delete a range. To change a range, add a new one and delete the old one.

### To edit an IP match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **IP addresses**.
3. In the **IP match conditions** pane, choose the IP match condition that you want to edit.
4. To add an IP address range:

   a. In the right pane, choose **Add IP address or range**.

   b. Select the correct IP version and type an IP address range by using CIDR notation. Here are some examples:

- To specify the IPv4 address 192.0.2.44, type **192.0.2.44/32**.
- To specify the IPv6 address 0:0:0:0:0:ffff:c000:22c, type **0:0:0:0:0:ffff:c000:22c/128**.
- To specify the range of IPv4 addresses from 192.0.2.0 to 192.0.2.255, type **192.0.2.0/24**.
- To specify the range of IPv6 addresses from 2620:0:2d0:200:0:0:0:0 to 2620:0:2d0:200:ffff:ffff:ffff:ffff, type **2620:0:2d0:200::/64**.

AWS WAF supports /8, /16, /24, and /32 IPv4 address ranges and /16, /24, /32, /56, /64, and /128 IPv6 address ranges. For more information about CIDR notation, see the Wikipedia entry Classless Inter-Domain Routing.

c. To add more IP addresses, choose **Add another IP address** and type the value.

d. Choose **Add**.

5. To delete an IP address or range:

   a. In the right pane, select the values that you want to delete.

   b. Choose **Delete IP address or range**.

## Deleting IP Match Conditions

If you want to delete an IP match condition, you must first delete all IP addresses and ranges in the condition and remove the condition from all the rules that are using it, as described in the following procedure.

### To delete an IP match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **IP addresses**.
3. In the **IP match conditions** pane, choose the IP match condition that you want to delete.
4. In the right pane, choose the **Rules** tab.

   If the list of rules using this IP match condition is empty, go to step 6. If the list contains any rules, make note of the rules, and continue with step 5.
5. To remove the IP match condition from the rules that are using it, perform the following steps:

   a. In the navigation pane, choose **Rules**.

   b. Choose the name of a rule that is using the IP match condition that you want to delete.

   c. In the right pane, select the IP match condition that you want to remove from the rule, and choose **Remove selected condition**.

   d. Repeat steps b and c for all the remaining rules that are using the IP match condition that you want to delete.

   e. In the navigation pane, choose **IP match conditions**.

   f. In the **IP match conditions** pane, choose the IP match condition that you want to delete.
6. Choose **Delete** to delete the selected condition.

## Working with Size Constraint Conditions

If you want to allow or block web requests based on the length of specified parts of requests, create one or more size constraint conditions. A size constraint condition identifies the part of web requests that you want AWS WAF to look at, the number of bytes that you want AWS WAF to look for, and an operator, such as greater than (>) or less than (<). For example, you can use a size constraint condition to look for query strings that are longer than 100 bytes. Later in the process, when you create a web ACL, you specify whether to allow or block requests based on those settings.

Note that if you configure AWS WAF to inspect the request body, for example, by searching the body for a specified string, AWS WAF inspects only the first 8192 bytes (8 KB). If the request body for your web requests will never exceed 8192 bytes, you can create a size constraint condition and block requests that have a request body greater than 8192 bytes.

Topics

## Creating Size Constraint Conditions

When you create size constraint conditions, you specify filters that identify the part of web requests for which you want AWS WAF to evaluate the length. You can add more than one filter to a size constraint condition, or you can create a separate condition for each filter. Here's how each configuration affects AWS WAF behavior:

- **One filter per size constraint condition** – When you add the separate size constraint conditions to a rule and add the rule to a web ACL, web requests must match all the conditions for AWS WAF to allow or block requests based on the conditions.

  For example, suppose you create two conditions. One matches web requests for which query strings are greater than 100 bytes. The other matches web requests for which the request body is greater than 1024 bytes. When you add both conditions to the same rule and add the rule to a web ACL, AWS WAF allows or blocks requests only when both conditions are true.
- **More than one filter per size constraint condition** – When you add a size constraint condition that contains multiple filters to a rule and add the rule to a web ACL, a web request needs only to match one of the filters in the size constraint condition for AWS WAF to allow or block the request based on that condition.

  Suppose you create one condition instead of two, and the one condition contains the same two filters as in the preceding example. AWS WAF allows or blocks requests if either the query string is greater than 100 bytes or the request body is greater than 1024 bytes.

> **Note**
> When you add a size constraint condition to a rule, you also can configure AWS WAF to allow or block web requests that *do not* match the values in the condition.

### To create a size constraint condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Size constraints**.
3. Choose **Create condition**.
4. Specify the applicable filter settings. For more information, see Values That You Specify When You Create or Edit Size Constraint Conditions (p. 39).
5. Choose **Add another filter**.
6. If you want to add another filter, repeat steps 4 and 5.

7. When you're finished adding filters, choose **Create size constraint condition**.

## Values That You Specify When You Create or Edit Size Constraint Conditions

When you create or update a size constraint condition, you specify the following values:

**Name**

Type a name for the size constraint condition.

The name can contain only the characters A-Z, a-z, and 0-9. You can't change the name of a condition after you create it.

**Part of the request to filter on**

Choose the part of each web request for which you want AWS WAF to evaluate the length:

**Header**

A specified request header, for example, the `User-Agent` or `Referer` header. If you choose **Header**, specify the name of the header in the **Header** field.

**HTTP method**

The HTTP method, which indicates the type of operation that the request is asking the origin to perform. CloudFront supports the following methods: `DELETE`, `GET`, `HEAD`, `OPTIONS`, `PATCH`, `POST`, and `PUT`.

**Query string**

The part of a URL that appears after a `?` character, if any.

**URI**

The part of a URL that identifies a resource, for example, `/images/daily-ad.jpg`.

**Body**

The part of a request that contains any additional data that you want to send to your web server as the HTTP request body, such as data from a form.

**Header (Only When "Part of the request to filter on" is "Header")**

If you chose **Header** for **Part of the request to filter on**, choose a header from the list of common headers, or type the name of a header for which you want AWS WAF to evaluate the length.

**Comparison operator**

Choose how you want AWS WAF to evaluate the length of the query string in web requests with respect to the value that you specify for **Size**.

For example, if you choose **Is greater than** for **Comparison operator** and type **100** for **Size**, AWS WAF evaluates web requests for a query string that is longer than 100 bytes.

**Size**

Type the length, in bytes, that you want AWS WAF to watch for in query strings.

**Note**
If you choose **URI** for the value of **Part of the request to filter on**, the **/** in the URI counts as one character. For example, the URI `/logo.jpg` is nine characters long.

### Transformation

A transformation reformats a web request before AWS WAF evaluates the length of the specified part of the request. This eliminates some of the unusual formatting that attackers use in web requests in an effort to bypass AWS WAF.

> **Note**
>
> If you choose **Body** for **Part of the request to filter on**, you can't configure AWS WAF to perform a transformation because only the first 8192 bytes are forwarded for inspection. However, you can still filter your traffic based on the size of the HTTP request body and specify a transformation of **None**. (AWS WAF gets the length of the body from the request headers.)

Transformations can perform the following operations:

**None**

AWS WAF doesn't perform any text transformations on the web request before checking the length.

**Convert to lowercase**

AWS WAF converts uppercase letters (A-Z) to lowercase (a-z).

**HTML decode**

AWS WAF replaces HTML-encoded characters with unencoded characters:

- Replaces `&quot;` with `&`
- Replaces `&nbsp;` with a non-breaking space
- Replaces `&lt;` with `<`
- Replaces `&gt;` with `>`
- Replaces characters that are represented in hexadecimal format, `&#xhhhh;`, with the corresponding characters
- Replaces characters that are represented in decimal format, `&#nnnn;`, with the corresponding characters

**Normalize whitespace**

AWS WAF replaces the following characters with a space character (decimal 32):

- \f, formfeed, decimal 12
- \t, tab, decimal 9
- \n, newline, decimal 10
- \r, carriage return, decimal 13
- \v, vertical tab, decimal 11
- non-breaking space, decimal 160

In addition, this option replaces multiple spaces with one space.

**Simplify command line**

For requests that contain operating system command line commands, use this option to perform the following transformations:

- Delete the following characters: \ " ' ^
- Delete spaces before the following characters: / (
- Replace the following characters with a space: , ;
- Replace multiple spaces with one space
- Convert uppercase letters (A-Z) to lowercase (a-z)

**URL decode**

Decode a URL-encoded request.

## Adding and Deleting Filters in a Size Constraint Condition

You can add or delete filters in a size constraint condition. To change a filter, add a new one and delete the old one.

### To add or delete filters in a size constraint condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Size constraint**.
3. Choose the condition that you want to add or delete filters in.
4. To add filters, perform the following steps:

    a. Choose **Add filter**.
    b. Specify the applicable filter settings. For more information, see Values That You Specify When You Create or Edit Size Constraint Conditions (p. 39).
    c. Choose **Add**.
5. To delete filters, perform the following steps:

    a. Select the filter that you want to delete.
    b. Choose **Delete filter**.

## Deleting Size Constraint Conditions

If you want to delete a size constraint condition, you need to first delete all filters in the condition and remove the condition from all the rules that are using it, as described in the following procedure.

### To delete a size constraint condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Size constraints**.
3. In the **Size constraint conditions** pane, choose the size constraint condition that you want to delete.
4. In the right pane, choose the **Associated rules** tab.

    If the list of rules using this size constraint condition is empty, go to step 6. If the list contains any rules, make note of the rules, and continue with step 5.
5. To remove the size constraint condition from the rules that are using it, perform the following steps:

    a. In the navigation pane, choose **Rules**.
    b. Choose the name of a rule that is using the size constraint condition that you want to delete.
    c. In the right pane, select the size constraint condition that you want to remove from the rule, and then choose **Remove selected condition**.
    d. Repeat steps b and c for all the remaining rules that are using the size constraint condition that you want to delete.
    e. In the navigation pane, choose **Size constraint**.
    f. In the **Size constraint conditions** pane, choose the size constraint condition that you want to delete.

6. Choose **Delete** to delete the selected condition.

# Working with SQL Injection Match Conditions

Attackers sometimes insert malicious SQL code into web requests in an effort to extract data from your database. To allow or block web requests that appear to contain malicious SQL code, create one or more SQL injection match conditions. A SQL injection match condition identifies the part of web requests, such as the URI or the query string, that you want AWS WAF to inspect. Later in the process, when you create a web ACL, you specify whether to allow or block requests that appear to contain malicious SQL code.

Topics

## Creating SQL Injection Match Conditions

When you create SQL injection match conditions, you specify filters, which indicate the part of web requests that you want AWS WAF to inspect for malicious SQL code, such as the URI or the query string. You can add more than one filter to a SQL injection match condition, or you can create a separate condition for each filter. Here's how each configuration affects AWS WAF behavior:

- **More than one filter per SQL injection match condition (recommended)** – When you add a SQL injection match condition containing multiple filters to a rule and add the rule to a web ACL, a web request needs only to match one of the filters in the SQL injection match condition for AWS WAF to allow or block the request based on that condition.

  For example, suppose you create one SQL injection match condition, and the condition contains two filters. One filter instructs AWS WAF to inspect the URI for malicious SQL code, and the other instructs AWS WAF to inspect the query string. AWS WAF allows or blocks requests if they appear to contain malicious SQL code *either* in the URI *or* in the query string.
- **One filter per SQL injection match condition** – When you add the separate SQL injection match conditions to a rule and add the rule to a web ACL, web requests must match all the conditions for AWS WAF to allow or block requests based on the conditions.

  Suppose you create two conditions, and each condition contains one of the two filters in the preceding example. When you add both conditions to the same rule and add the rule to a web ACL, AWS WAF allows or blocks requests only when both the URI and the query string appear to contain malicious SQL code.

> **Note**
> When you add a SQL injection match condition to a rule, you also can configure AWS WAF to allow or block web requests that *do not* appear to contain malicious SQL code.

### To create a SQL injection match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **SQL injection**.
3. Choose **Create condition**.
4. Specify the applicable filter settings. For more information, see Values That You Specify When You Create or Edit SQL Injection Match Conditions (p. 43).

5. Choose **Add another filter**.
6. If you want to add another filter, repeat steps 4 and 5.
7. When you're finished adding filters, choose **Create**.

## Values That You Specify When You Create or Edit SQL Injection Match Conditions

When you create or update a SQL injection match condition, you specify the following values:

**Name**

The name of the SQL injection match condition.

The name can contain only the characters A-Z, a-z, and 0-9. You can't change the name of a condition after you create it.

**Part of the request to filter on**

Choose the part of each web request that you want AWS WAF to inspect for malicious SQL code:

**Header**

A specified request header, for example, the `User-Agent` or `Referer` header. If you choose **Header**, specify the name of the header in the **Header** field.

**HTTP method**

The HTTP method, which indicates the type of operation that the request is asking the origin to perform. CloudFront supports the following methods: `DELETE`, `GET`, `HEAD`, `OPTIONS`, `PATCH`, `POST`, and `PUT`.

**Query string**

The part of a URL that appears after a `?` character, if any.

**URI**

The part of a URL that identifies a resource, for example, `/images/daily-ad.jpg`.

**Body**

The part of a request that contains any additional data that you want to send to your web server as the HTTP request body, such as data from a form.

> **Note**
> If you choose **Body** for the value of **Part of the request to filter on**, AWS WAF inspects only the first 8192 bytes (8 KB). To allow or block requests for which the body is longer than 8192 bytes, you can create a size constraint condition. (AWS WAF gets the length of the body from the request headers.) For more information, see Working with Size Constraint Conditions (p. 37).

**Header**

If you chose **Header** for **Part of the request to filter on**, choose a header from the list of common headers, or type the name of a header that you want AWS WAF to inspect for malicious SQL code.

**Transformation**

A transformation reformats a web request before AWS WAF inspects the request. This eliminates some of the unusual formatting that attackers use in web requests in an effort to bypass AWS WAF. Transformations can perform the following operations:

**None**

AWS WAF doesn't perform any text transformations on the web request before inspecting it for the string in **Value to match**.

**Convert to lowercase**

AWS WAF converts uppercase letters (A-Z) to lowercase (a-z).

**HTML decode**

AWS WAF replaces HTML-encoded characters with unencoded characters:

- Replaces `&quot;` with `&`
- Replaces `&nbsp;` with a non-breaking space
- Replaces `&lt;` with `<`
- Replaces `&gt;` with `>`
- Replaces characters that are represented in hexadecimal format, `&#xhhhh;`, with the corresponding characters
- Replaces characters that are represented in decimal format, `&#nnnn;`, with the corresponding characters

**Normalize whitespace**

AWS WAF replaces the following characters with a space character (decimal 32):

- \f, formfeed, decimal 12
- \t, tab, decimal 9
- \n, newline, decimal 10
- \r, carriage return, decimal 13
- \v, vertical tab, decimal 11
- non-breaking space, decimal 160

In addition, this option replaces multiple spaces with one space.

**Simplify command line**

For requests that contain operating system command line commands, use this option to perform the following transformations:

- Delete the following characters: \ " ' ^
- Delete spaces before the following characters: / (
- Replace the following characters with a space: , ;
- Replace multiple spaces with one space
- Convert uppercase letters (A-Z) to lowercase (a-z)

**URL decode**

Decode a URL-encoded request.

## Adding and Deleting Filters in a SQL Injection Match Condition

You can add or delete filters in a SQL injection match condition. To change a filter, add a new one and delete the old one.

### To add or delete filters in a SQL injection match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.

2. In the navigation pane, choose **SQL injection**.
3. Choose the condition that you want to add or delete filters in.
4. To add filters, perform the following steps:

   a. Choose **Add filter**.

   b. Specify the applicable filter settings. For more information, see Values That You Specify When You Create or Edit SQL Injection Match Conditions (p. 43).

   c. Choose **Add**.
5. To delete filters, perform the following steps:

   a. Select the filter that you want to delete.

   b. Choose **Delete filter**.

## Deleting SQL Injection Match Conditions

If you want to delete a SQL injection match condition, you need to first delete all filters in the condition and remove the condition from all the rules that are using it, as described in the following procedure.

### To delete a SQL injection match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **SQL injection**.
3. In the **SQL injection match conditions** pane, choose the SQL injection match condition that you want to delete.
4. In the right pane, choose the **Associated rules** tab.

   If the list of rules using this SQL injection match condition is empty, go to step 6. If the list contains any rules, make note of the rules, and continue with step 5.
5. To remove the SQL injection match condition from the rules that are using it, perform the following steps:

   a. In the navigation pane, choose **Rules**.

   b. Choose the name of a rule that is using the SQL injection match condition that you want to delete.

   c. In the right pane, select the SQL injection match condition that you want to remove from the rule, and choose **Remove selected condition**.

   d. Repeat steps b and c for all of the remaining rules that are using the SQL injection match condition that you want to delete.

   e. In the navigation pane, choose **SQL injection**.

   f. In the **SQL injection match conditions** pane, choose the SQL injection match condition that you want to delete.
6. Choose **Delete** to delete the selected condition.

# Working with String Match Conditions

If you want to allow or block web requests based on strings that appear in the requests, create one or more string match conditions. A string match condition identifies the string that you want to search for and the part of web requests, such as a specified header or the query string, that you want AWS WAF to inspect for the string. Later in the process, when you create a web ACL, you specify whether to allow or block requests that contain the string.

Topics

## Creating a String Match Condition

When you create string match conditions, you specify filters that identify the string that you want to search for and the part of web requests that you want AWS WAF to inspect for that string, such as the URI or the query string. You can add more than one filter to a string match condition, or you can create a separate string match condition for each filter. Here's how each configuration affects AWS WAF behavior:

- **One filter per string match condition** – When you add the separate string match conditions to a rule and add the rule to a web ACL, web requests must match all the conditions for AWS WAF to allow or block requests based on the conditions.

  For example, suppose you create two conditions. One matches web requests that contain the value `BadBot` in the `User-Agent` header. The other matches web requests that contain the value `BadParameter` in query strings. When you add both conditions to the same rule and add the rule to a web ACL, AWS WAF allows or blocks requests only when they contain both values.
- **More than one filter per string match condition** – When you add a string match condition that contains multiple filters to a rule and add the rule to a web ACL, a web request needs only to match one of the filters in the string match condition for AWS WAF to allow or block the request based on the one condition.

  Suppose you create one condition instead of two, and the one condition contains the same two filters as in the preceding example. AWS WAF allows or blocks requests if they contain *either* `BadBot` in the `User-Agent` header *or* `BadParameter` in the query string.

> **Note**
> When you add a string match condition to a rule, you also can configure AWS WAF to allow or block web requests that *do not* match the values in the condition.

**To create a string match condition**

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **String matching**.
3. Choose **Create condition**.
4. Specify the applicable filter settings. For more information, see Values That You Specify When You Create or Edit String Match Conditions (p. 46).
5. Choose **Add another filter**.
6. If you want to add another filter, repeat steps 4 and 5.
7. When you're finished adding filters, choose **Create string match condition**.

## Values That You Specify When You Create or Edit String Match Conditions

When you create or update a string match condition, you specify the following values:

**Name**

Type a name for the string match condition. The value can contain only the characters A-Z, a-z, and 0-9. You can't change the name of a condition after you create it.

**Part of the request to filter on**

Choose the part of each web request that you want AWS WAF to inspect for the string that you specify in **Value to match**:

**Header**

A specified request header, for example, the `User-Agent` or `Referer` header. If you choose **Header**, specify the name of the header in the **Header** field.

**HTTP method**

The HTTP method, which indicates the type of operation that the request is asking the origin to perform. CloudFront supports the following methods: `DELETE`, `GET`, `HEAD`, `OPTIONS`, `PATCH`, `POST`, and `PUT`.

**Query string**

The part of a URL that appears after a `?` character, if any.

**URI**

The part of a URL that identifies a resource, for example, `/images/daily-ad.jpg`.

**Body**

The part of a request that contains any additional data that you want to send to your web server as the HTTP request body, such as data from a form.

> **Note**
> If you choose **Body** for the value of **Part of the request to filter on**, AWS WAF inspects only the first 8192 bytes (8 KB). To allow or block requests for which the body is longer than 8192 bytes, you can create a size constraint condition. (AWS WAF gets the length of the body from the request headers.) For more information, see Working with Size Constraint Conditions (p. 37).

**Header (Only When "Part of the request to filter on" is "Header")**

If you chose **Header** from the **Part of the request to filter on** list, choose a header from the list of common headers, or type the name of a header that you want AWS WAF to inspect.

**Match type**

Within the part of the request that you want AWS WAF to inspect, choose where the string in **Value to match** must appear to match this filter:

**Contains**

The string appears anywhere in the specified part of the request.

**Contains word**

The specified part of the web request must include **Value to match**, and **Value to match** must contain only alphanumeric characters or underscore (A-Z, a-z, 0-9, or _). In addition, **Value to match** must be a word, which means one of the following:

- **Value to match** exactly matches the value of the specified part of the web request, such as the value of a header.
- **Value to match** is at the beginning of the specified part of the web request and is followed by a character other than an alphanumeric character or underscore (_), for example, `BadBot;`.
- **Value to match** is at the end of the specified part of the web request and is preceded by a character other than an alphanumeric character or underscore (_), for example, `;BadBot`.

- **Value to match** is in the middle of the specified part of the web request and is preceded and followed by characters other than alphanumeric characters or underscore (_), for example, –`BadBot;`.

**Exactly matches**

The string and the value of the specified part of the request are identical.

**Starts with**

The string appears at the beginning of the specified part of the request.

**Ends with**

The string appears at the end of the specified part of the request.

**Transformation**

A transformation reformats a web request before AWS WAF inspects the request. This eliminates some of the unusual formatting that attackers use in web requests in an effort to bypass AWS WAF. Transformations can perform the following operations:

**None**

AWS WAF doesn't perform any text transformations on the web request before inspecting it for the string in **Value to match**.

**Convert to lowercase**

AWS WAF converts uppercase letters (A-Z) to lowercase (a-z).

**HTML decode**

AWS WAF replaces HTML-encoded characters with unencoded characters:

- Replaces `&quot;` with `&`
- Replaces `&nbsp;` with a non-breaking space
- Replaces `&lt;` with `<`
- Replaces `&gt;` with `>`
- Replaces characters that are represented in hexadecimal format, `&#xhhhh;`, with the corresponding characters
- Replaces characters that are represented in decimal format, `&#nnnn;`, with the corresponding characters

**Normalize whitespace**

AWS WAF replaces the following characters with a space character (decimal 32):

- \f, formfeed, decimal 12
- \t, tab, decimal 9
- \n, newline, decimal 10
- \r, carriage return, decimal 13
- \v, vertical tab, decimal 11
- non-breaking space, decimal 160

In addition, this option replaces multiple spaces with one space.

**Simplify command line**

When you're concerned that attackers are injecting an operating system command line command and using unusual formatting to disguise some or all of the command, use this option to perform the following transformations:

- Delete the following characters: \ " ' ^
- Delete spaces before the following characters: / (
- Replace the following characters with a space: , ;
- Replace multiple spaces with one space
- Convert uppercase letters (A-Z) to lowercase (a-z)

**URL decode**

Decode a URL-encoded request.

**Value is base64 encoded**

If the value in **Value to match** is base64-encoded, select this check box. Use base64-encoding to specify non-printable characters, such as tabs and linefeeds, that attackers include in their requests.

**Value to match**

Specify the value that you want AWS WAF to search for in web requests. The maximum length is 50 bytes. If you're base64-encoding the value, the 50-byte limit applies to the value before you encode it.

## Adding and Deleting Filters in a String Match Condition

You can add filters to a string match condition or delete filters. To change a filter, add a new one and delete the old one.

### To add or delete filters in a string match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **String matching**.
3. Choose the condition that you want to add or delete filters in.
4. To add filters, perform the following steps:

   a. Choose **Add filter**.
   b. Specify the applicable filter settings. For more information, see Values That You Specify When You Create or Edit String Match Conditions (p. 46).
   c. Choose **Add**.
5. To delete filters, perform the following steps:

   a. Select the filter that you want to delete.
   b. Choose **Delete Filter**.

## Deleting String Match Conditions

If you want to delete a string match condition, you need to first delete all filters in the condition and remove the condition from all the rules that are using it, as described in the following procedure.

### To delete a string match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **String matching**.
3. In the **String match conditions** pane, choose the string match condition that you want to delete.

4. In the right pane, choose the **Rules** tab.

   If the list of rules using this string match condition is empty, go to step 6. If the list contains any rules, make note of the rules, and continue with step 5.
5. To remove the string match condition from the rules that are using it, perform the following steps:

   a. In the navigation pane, choose **Rules**.
   b. Choose the name of a rule that is using the string match condition that you want to delete.
   c. In the right pane, select the string match condition that you want to remove from the rule, and then choose **Remove selected condition**.
   d. Repeat steps b and c for all the remaining rules that are using the string match condition that you want to delete.
   e. In the navigation pane, choose **String match**.
   f. In the **String match conditions** pane, choose the string match condition that you want to delete.
6. Choose **Delete** to delete the selected condition.

# Working with Rules

Rules let you precisely target the web requests that you want AWS WAF to allow or block by specifying the exact conditions that you want AWS WAF to watch for. For example, AWS WAF can watch for the IP addresses that requests originate from, the strings that the requests contain and where the strings appear, and whether the requests appear to contain malicious SQL code.

Topics

## Creating a Rule and Adding Conditions

If you add more than one condition to a rule, a web request must match all the conditions for AWS WAF to allow or block requests based on that rule.

### To create a rule and add conditions

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Rules**.
3. Choose **Create rule**.
4. Type the following values:

   **Name**

   Type a name.

   **CloudWatch metric name**

   Type a name for the CloudWatch metric that AWS WAF will create and will associate with the rule. The name can contain only alphanumeric characters (A-Z, a-z, 0-9). It can't contain whitespace.

**Rule type**

Choose either `Regular rule` or `Rate-based rule`. Rate–based rules are identical to regular rules, but also take into account how many requests arrive from a specified IP address every five minutes. For more information about these rule types, see How AWS WAF Works (p. 6).

**Rate limit**

If you are creating a rate–based rule, enter the maximum number of requests from a single IP address allowed in a five-minute period. The rate limit must be equal to or greater than 2000.

5. To add a condition to the rule, specify the following values:

   **When a request does/does not**

   If you want AWS WAF to allow or block requests based on the filters in a condition, for example, web requests that originate from the range of IP addresses 192.0.2.0/24, choose **does**.

   If you want AWS WAF to allow or block requests based on the inverse of the filters in a condition, choose **does not**. For example, if an IP match condition includes the IP address range 192.0.2.0/24 and you want AWS WAF to allow or block requests that *do not* come from those IP addresses, choose **does not**.

   **match/originate from**

   Choose the type of condition that you want to add to the rule:

   - Cross-site scripting match conditions – choose **match at least one of the filters in the cross-site scripting match condition**
   - IP match conditions – choose **originate from an IP address in**
   - Size constraint conditions – choose **match at least one of the filters in the size constraint condition**
   - SQL injection match conditions – choose **match at least one of the filters in the SQL injection match condition**
   - String match conditions – choose **match at least one of the filters in the string match condition**

   **condition name**

   Choose the condition that you want to add to the rule. The list displays only conditions of the type that you chose in the preceding step.

6. To add another condition to the rule, choose **Add another condition**, and repeat steps 4 and 5. Note the following:

   - If you add more than one condition, a web request must match at least one filter in every condition for AWS WAF to allow or block requests based on that rule
   - If you add two IP match conditions to the same rule, AWS WAF will only allow or block requests that originate from IP addresses that appear in both IP match conditions

7. When you're finished adding conditions, choose **Create**.

## Adding and Removing Conditions in a Rule

You can change a rule by adding or removing conditions.

### To add or remove conditions in a rule

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Rules**.

3. Choose the rule in which you want to add or remove conditions.
4. To add a condition, choose **Add condition** and specify the following values:

   **When a request does/does not**

   If you want AWS WAF to allow or block requests based on the filters in a condition, for example, web requests that originate from the range of IP addresses 192.0.2.0/24, choose **does**.

   If you want AWS WAF to allow or block requests based on the inverse of the filters in a condition, choose **does not**. For example, if an IP match condition includes the IP address range 192.0.2.0/24 and you want AWS WAF to allow or block requests that *do not* come from those IP addresses, choose **does not**.

   **match/originate from**

   Choose the type of condition that you want to add to the rule:

   - Cross-site scripting match conditions – choose **match at least one of the filters in the cross-site scripting match condition**
   - IP match conditions – choose **originate from an IP address in**
   - Size constraint conditions – choose **match at least one of the filters in the size constraint condition**
   - SQL injection match conditions – choose **match at least one of the filters in the SQL injection match condition**
   - String match conditions – choose **match at least one of the filters in the string match condition**

   ***condition name***

   Choose the condition that you want to add to the rule. The list displays only conditions of the type that you chose in the preceding step.
5. To remove a condition, select the condition, and then choose **Remove selected condition**.

## Deleting a Rule

If you want to delete a rule, you need to first remove the rule from the web ACLs that are using it and remove the conditions that are included in the rule.

### To delete a rule

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Rules**.
3. In the **Rules** pane, choose the rule that you want to delete.
4. In the right pane, choose the **Web ACLs** tab.

   If the list of the web ACLs that are using this rule is empty, go to step 6. If the list contains any web ACLs, make note of them, and continue with step 5.
5. To remove the rule from the web ACLs that are using it, perform the following steps:

   a. In the navigation pane, choose **Web ACLs**.
   b. Choose the name of a web ACL that is using the rule that you want to delete.
   c. In the right pane, select the rule that you want to remove from the web ACL, and then choose **Remove selected rule**.
   d. Repeat steps b and c for all of the remaining web ACLs that are using the rule that you want to delete.

   e. In the navigation pane, choose **Rules**.
   f. In the **Rules** pane, choose the rule that you want to delete.

6. To delete the selected rule, choose **Actions**, and then choose **Delete**.

## Listing the Web ACLs That Include a Specified Rule

If you want to know which web ACLs will be affected if you edit or delete a rule, perform the following procedure.

**To list the web ACLs that include a rule**

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Rules**.
3. In the **Rules** pane, choose the rule for which you want to find the associated web ACLs.
4. In the right pane, choose the **Web ACLs** tab.

# Working with Web ACLs

When you add rules to a web ACL, you specify whether you want AWS WAF to allow or block requests based on the conditions in the rules. If you add more than one rule to a web ACL, AWS WAF evaluates each request against the rules in the order that you list them in the web ACL. When a web request matches all the conditions in a rule, AWS WAF immediately takes the corresponding action—allow or block—and doesn't evaluate the request against the remaining rules in the web ACL, if any.

If a web request doesn't match any of the rules in a web ACL, AWS WAF takes the default action that you specified for the web ACL. For more information, see Deciding on the Default Action for a Web ACL (p. 31).

If you want to test a rule before you start using it to allow or block requests, you can configure AWS WAF to count the web requests that match the conditions in the rule. For more information, see Testing Web ACLs (p. 57).

Topics

## Creating a Web ACL

**To create a web ACL**

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. If this is your first time using AWS WAF, choose **Go to AWS WAF** and then **Configure Web ACL**. If you've used AWS WAF before, choose **Web ACLs** in the navigation pane, and then choose **Create web ACL**.
3. For **Web ACL name**, type a name.

> **Note**
> You can't change the name after you create the web ACL.

4. For **CloudWatch metric name**, change the default name if applicable. The name can contain only alphanumeric characters (A-Z, a-z, 0-9). It can't contain whitespace.

   > **Note**
   > You can't change the name after you create the web ACL.

5. For **Region**, choose a region.
6. For **AWS resource**, choose the resource that you want to associate with this web ACL, and then choose **Next**.
7. If you've already created the conditions that you want AWS WAF to use to inspect your web requests, choose **Next**, and then continue to the next step.

   If you haven't already created conditions, do so now. For more information, see the following topics:

   - Working with Cross-site Scripting Match Conditions (p. 31)
   - Working with IP Match Conditions (p. 35)
   - Working with Size Constraint Conditions (p. 37)
   - Working with SQL Injection Match Conditions (p. 42)
   - Working with String Match Conditions (p. 45)
8. If you've already created the rules that you want to add to this web ACL, add the rules to the web ACL:

   a. In the **Rules** list, choose a rule.
   b. Choose **Add rule to web ACL**.
   c. Repeat steps a and b until you've added all the rules that you want to add to this web ACL.
   d. Go to step 10.
9. If you haven't created rules yet, you can add rules now:

   a. Choose **Create rule**.
   b. Type the following values:

      **Name**

      Type a name.

      **CloudWatch metric name**

      Type a name for the CloudWatch metric that AWS WAF will create and will associate with the rule. The name can contain only alphanumeric characters (A-Z, a-z, 0-9). It can't contain whitespace.

      > **Note**
      > You can't change the metric name after you create the rule.
   c. To add a condition to the rule, specify the following values:

      **When a request does/does not**

      If you want AWS WAF to allow or block requests based on the filters in a condition, for example, web requests that originate from the range of IP addresses 192.0.2.0/24, choose **does**.

      If you want AWS WAF to allow or block requests based on the inverse of the filters in a condition, choose **does not**. For example, if an IP match condition includes the IP address range 192.0.2.0/24 and you want AWS WAF to allow or block requests that *do not* come from those IP addresses, choose **does not**.

**match/originate from**

Choose the type of condition that you want to add to the rule:

- Cross-site scripting match conditions – choose **match at least one of the filters in the cross-site scripting match condition**
- IP match conditions – choose **originate from an IP address in**
- Size constraint conditions – choose **match at least one of the filters in the size constraint condition**
- SQL injection match conditions – choose **match at least one of the filters in the SQL injection match condition**
- String match conditions – choose **match at least one of the filters in the string match condition**

**condition name**

Choose the condition that you want to add to the rule. The list displays only conditions of the type that you chose in the preceding list.

d. To add another condition to the rule, choose **Add another condition**, and then repeat steps b and c. Note the following:

- If you add more than one condition, a web request must match at least one filter in every condition for AWS WAF to allow or block requests based on that rule
- If you add two IP match conditions to the same rule, AWS WAF will only allow or block requests that originate from IP addresses that appear in both IP match conditions

e. Repeat step 9 until you've created all the rules that you want to add to this web ACL.

f. Choose **Create**.

g. Continue with step 10.

10. For each rule that you've added to the web ACL, choose whether you want AWS WAF to allow, block, or count web requests based on the conditions in the rule:

    - **Allow** – CloudFront or an Application Load Balancer responds with the requested object. In the case of CloudFront, if the object isn't in the edge cache, CloudFront forwards the request to the origin.
    - **Block** – CloudFront or an Application Load Balancer responds to the request with an HTTP 403 (Forbidden) status code. CloudFront also can respond with a custom error page. For more information, see Using AWS WAF with CloudFront Custom Error Pages (p. 61).
    - **Count** – AWS WAF increments a counter of requests that match the conditions in the rule, and then continues to inspect the web request based on the remaining rules in the web ACL.

      For information about using **Count** to test a web ACL before you start to use it to allow or block web requests, see Counting the Web Requests That Match the Rules in a Web ACL (p. 57).
11. If you want to change the order of the rules in the web ACL, use the arrows in the **Order** column. AWS WAF inspects web requests based on the order in which rules appear in the web ACL.
12. If you want to remove a rule that you added to the web ACL, choose the **x** in the row for the rule.
13. Choose the default action for the web ACL. This is the action that AWS WAF takes when a web request doesn't match the conditions in any of the rules in this web ACL. For more information, see Deciding on the Default Action for a Web ACL (p. 31).
14. Choose **Review and create**.
15. Review the settings for the web ACL, and choose **Confirm and create**.

## Associating or Disassociating a Web ACL with a CloudFront Distribution or an Application Load Balancer

To associate or disassociate a web ACL, perform the applicable procedure. Note that you also can associate a web ACL with a CloudFront distribution when you create or update the distribution. For more information, see Using AWS WAF to Control Access to Your Content in the *Amazon CloudFront Developer Guide*.

> **Note**
> You can associate a web ACL with as many CloudFront distributions or Application Load Balancers as you want, but you can associate only one web ACL with a given distribution.

### To associate a web ACL with a CloudFront distribution or Application Load Balancer

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Web ACLs**.
3. Choose the web ACL that you want to associate with a CloudFront distribution or Application Load Balancer.
4. On the **Rules** tab, under **AWS resources using this web ACL**, choose **Add association**.
5. When prompted, use the **Resource** list to choose the CloudFront distribution or Application Load Balancer that you want to associate this web ACL with. If you choose an Application Load Balancer, you also must specify a region.
6. Choose **Add**.
7. To associate this web ACL with additional CloudFront distributions or Application Load Balancers, repeat steps 4 through 6.

### To disassociate a web ACL from a CloudFront distribution or Application Load Balancer

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Web ACLs**.
3. Choose the web ACL that you want to disassociate from a CloudFront distribution or Application Load Balancer.
4. On the **Rules** tab, under **AWS resources using this web ACL**, choose the **x** for each CloudFront distribution or Application Load Balancer that you want to disassociate this web ACL from.

## Editing a Web ACL

To add or remove rules from a web ACL or change the default action, perform the following procedure.

### To edit a web ACL

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Web ACLs**.
3. Choose the web ACL that you want to edit.
4. On the **Rules** tab in the right pane, choose **Edit web ACL**.
5. To add rules to the web ACL, perform the following steps:

    a. In the **Rules** list, choose the rule that you want to add.

   b. Choose **Add rule to web ACL**.
   c. Repeat steps a and b until you've added all the rules that you want.
6. If you want to change the order of the rules in the web ACL, use the arrows in the **Order** column. AWS WAF inspects web requests based on the order in which rules appear in the web ACL.
7. To remove a rule from the web ACL, choose the **x** at the right of the row for that rule. This doesn't delete the rule from AWS WAF, it just removes the rule from this web ACL.
8. To change the action for a rule or the default action for the web ACL, choose the preferred option.
9. Choose **Save changes**.

## Deleting a Web ACL

To delete a web ACL, you must remove the rules that are included in the web ACL and disassociate all CloudFront distributions and Application Load Balancers from the web ACL. Perform the following procedure.

**To delete a web ACL**

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Web ACLs**.
3. Choose the web ACL that you want to delete.
4. On the **Rules** tab in the right pane, choose **Edit web ACL**.
5. To remove all rules from the web ACL, choose the **x** at the right of the row for each rule. This doesn't delete the rules from AWS WAF, it just removes the rules from this web ACL.
6. Choose **Update**.
7. Disassociate the web ACL from all CloudFront distributions and Application Load Balancers. On the **Rules** tab, under **AWS resources using this web ACL**, choose the **x** for each CloudFront distribution or Application Load Balancer.
8. On the **Web ACLs** page, confirm that the web ACL that you want to delete is selected, and then choose **Delete**.

# Testing Web ACLs

To ensure that you don't accidentally configure AWS WAF to block web requests that you want to allow or allow requests that you want to block, we recommend that you test your web ACL thoroughly before you start using it on your website or web application.

Topics

## Counting the Web Requests That Match the Rules in a Web ACL

When you add rules to a web ACL, you specify whether you want AWS WAF to allow, block, or count the web requests that match all the conditions in that rule. We recommend that you begin with the following configuration:

- Configure all the rules in a web ACL to count web requests
- Set the default action for the web ACL to allow requests

In this configuration, AWS WAF inspects each web request based on the conditions in the first rule. If the web request matches all the conditions in that rule, AWS WAF increments a counter for that rule. Then AWS WAF inspects the web request based on the conditions in the next rule. If the request matches all the conditions in that rule, AWS WAF increments a counter for the rule. This continues until AWS WAF has inspected the request based on the conditions in all of your rules.

After you've configured all the rules in a web ACL to count requests and associated the web ACL with a CloudFront distribution or Application Load Balancer, you can view the resulting counts in an Amazon CloudWatch graph. For each rule in a web ACL and for all the requests that CloudFront or an Application Load Balancer forwards to AWS WAF for a web ACL, CloudWatch lets you:

- View data for the preceding hour or preceding three hours,
- Change the interval between data points
- Change the calculation that CloudWatch performs on the data, such as maximum, minimum, average, or sum

> **Note**
> AWS WAF with CloudFront is a global service and metrics are available only when you choose the **US East (N. Virginia)** Region in the AWS console. If you choose another region, no AWS WAF metrics will appear in the CloudWatch console.

### To view data for the rules in a web ACL

1. Sign in to the AWS Management Console and open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, under **Metrics**, choose **WAF**.
3. Select the check box for the web ACL that you want to view data for.
4. Change the applicable settings:

   **Statistic**

   Choose the calculation that CloudWatch performs on the data.

   **Time range**

   Choose whether you want to view data for the preceding hour or the preceding three hours.

   **Period**

   Choose the interval between data points in the graph.

   **Rules**

   Choose the rules for which you want to view data.

   Note the following:

   - If you just associated a web ACL with a CloudFront distribution or Application Load Balancer, you might need to wait a few minutes for data to appear in the graph and for the metric for the web ACL to appear in the list of available metrics.
   - If you associate more than one CloudFront distribution or Application Load Balancer with a web ACL, the CloudWatch data will include all the requests for all the distributions that are associated with the web ACL.

- You can hover the mouse cursor over a data point to get more information.
- The graph doesn't refresh itself automatically. To update the display, choose the refresh (⟳) icon.

5. (Optional) View detailed information about individual requests that CloudFront or an Application Load Balancer has forwarded to AWS WAF. For more information, see Viewing a Sample of the Web Requests That CloudFront or an Application Load Balancer Has Forwarded to AWS WAF (p. 59).
6. If you determine that a rule is intercepting requests that you don't want it to intercept, change the applicable settings. For more information, see Creating and Configuring a Web Access Control List (Web ACL) (p. 30).

   When you're satisfied that all of your rules are intercepting only the correct requests, change the action for each of your rules to **Allow** or **Block**. For more information, see Editing a Web ACL (p. 56).

# Viewing a Sample of the Web Requests That CloudFront or an Application Load Balancer Has Forwarded to AWS WAF

In the AWS WAF console, you can view a sample of the requests that CloudFront or an Application Load Balancer has forwarded to AWS WAF for inspection. 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. You also can view which rule the request matched, and whether the rule is configured to allow or block requests.

The sample of requests contains up to 100 requests that matched all the conditions in each rule and another 100 requests for the default action, which applies to requests that didn't match all the conditions in any rule. The requests in the sample come from all the CloudFront edge locations or Application Load Balancers that have received requests for your content in the previous 15 minutes.

**To view a sample of the web requests that CloudFront or an Application Load Balancer has forwarded to AWS WAF**

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose the web ACL for which you want to view requests.
3. In the right pane, choose the **Requests** tab.

   The **Sampled requests** table displays the following values for each request:

   **Source IP**

   Either the IP address that the request originated from or, if the viewer used an HTTP proxy or an Application Load Balancer to send the request, the IP address of the proxy or Application Load Balancer.

   **URI**

   The part of a URL that identifies a resource, for example, `/images/daily-ad.jpg`.

   **Matches rule**

   Identifies the first rule in the web ACL for which the web request matched all the conditions. If a web request doesn't match all the conditions in any rule in the web ACL, the value of **Matches rule** is **Default**.

Note that when a web request matches all the conditions in a rule and the action for that rule is **Count**, AWS WAF continues inspecting the web request based on subsequent rules in the web ACL. In this case, a web request could appear twice in the list of sampled requests: once for the rule that has an action of **Count** and again for a subsequent rule or for the default action.

**Action**

Indicates whether the action for the corresponding rule is **Allow**, **Block**, or **Count**.

**Time**

The time that AWS WAF received the request from CloudFront or your Application Load Balancer.

4. To display additional information about the request, choose the arrow on the left side of the IP address for that request. AWS WAF displays the following information:

   **Source IP**

   The same IP address as the value in the **Source IP** column in the table.

   **Country**

   The two-letter country code of the country that the request originated from. If the viewer used an HTTP proxy or an Application Load Balancer to send the request, this is the two-letter country code of the country that the HTTP proxy or an Application Load Balancer is in.

   For a list of two-letter country codes and the corresponding country names, see the Wikipedia entry ISO 3166-1 alpha-2.

   **Method**

   The HTTP request method for the request: `GET`, `HEAD`, `OPTIONS`, `PUT`, `POST`, `PATCH`, or `DELETE`.

   **URI**

   The same URI as the value in the **URI** column in the table.

   **Request headers**

   The request headers and header values in the request.

5. To refresh the list of sample requests, choose **Get new samples**.

# Listing IP addresses blocked by rate-based rules

AWS WAF provides a list of IP addresses that are blocked by rate-based rules.

### To view addresses blocked by rate-based rules

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. In the navigation pane, choose **Rules**.
3. In the **Name** column, choose a rate-based rule.

   The list shows the IP addresses that the rule currently blocks.

# How AWS WAF Works with Amazon CloudFront Features

When you create a web ACL, you can specify one or more CloudFront distributions that you want AWS WAF to inspect. AWS WAF starts to allow, block, or count web requests for those distributions based on the conditions that you identify in the web ACL. CloudFront provides some features that enhance the AWS WAF functionality. This chapter describes a few ways that you can configure CloudFront to make CloudFront and AWS WAF work better together.

Topics

## Using AWS WAF with CloudFront Custom Error Pages

When AWS WAF blocks a web request based on the conditions that you specify, it returns HTTP status code 403 (Forbidden) to CloudFront. Next, CloudFront returns that status code to the viewer. The viewer then displays a brief and sparsely formatted default message similar to this:

```
Forbidden: You don't have permission to access /myfilename.html on this server.
```

If you'd rather display a custom error message, possibly using the same formatting as the rest of your website, you can configure CloudFront to return to the viewer an object (for example, an HTML file) that contains your custom error message.

> **Note**
> CloudFront can't distinguish between an HTTP status code 403 that is returned by your origin and one that is returned by AWS WAF when a request is blocked. This means that you can't return different custom error pages based on the different causes of an HTTP status code 403.

For more information about CloudFront custom error pages, see Customizing Error Responses in the *Amazon CloudFront Developer Guide*.

## Using AWS WAF with CloudFront Geo Restriction

You can use the Amazon CloudFront *geo restriction* feature, also known as *geoblocking*, to prevent users in specific geographic locations from accessing content that you distribute through a CloudFront web distribution. If you want to block web requests from specific countries and also block requests based on other conditions, you can use CloudFront geo restriction in conjunction with AWS WAF. CloudFront returns the same HTTP status code to viewers—HTTP 403 (Forbidden)—whether they try to access your content from a country on a CloudFront geo restriction blacklist or whether the request is blocked by AWS WAF.

> **Note**
> You can see the two-letter country code of the country that requests originate from in the sample of web requests for a web ACL. For more information, see Viewing a Sample of the Web Requests That CloudFront or an Application Load Balancer Has Forwarded to AWS WAF (p. 59).

For more information about CloudFront geo restriction, see Restricting the Geographic Distribution of Your Content in the *Amazon CloudFront Developer Guide*.

## Choosing the HTTP Methods That CloudFront Responds To

When you create an Amazon CloudFront web distribution, you choose the HTTP methods that you want CloudFront to process and forward to your origin. You can choose from the following options:

- **GET, HEAD** – You can use CloudFront only to get objects from your origin or to get object headers.
- **GET, HEAD, OPTIONS** – You can use CloudFront only to get objects from your origin, get object headers, or retrieve a list of the options that your origin server supports.
- **GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE** – You can use CloudFront to get, add, update, and delete objects, and to get object headers. In addition, you can perform other POST operations such as submitting data from a web form.

You also can use AWS WAF string match conditions to allow or block requests based on the HTTP method, as described in Working with String Match Conditions (p. 45). If you want to use a combination of methods that CloudFront supports, such as `GET` and `HEAD`, then you don't need to configure AWS WAF to block requests that use the other methods. If you want to allow a combination of methods that CloudFront doesn't support, such as `GET`, `HEAD`, and `POST`, you can configure CloudFront to respond to all methods, and then use AWS WAF to block requests that use other methods.

For more information about choosing the methods that CloudFront responds to, see Allowed HTTP Methods in the topic Values that You Specify When You Create or Update a Web Distribution in the *Amazon CloudFront Developer Guide*.

# Authentication and Access Control for AWS WAF

Access to AWS WAF requires credentials. Those credentials must have permissions to access AWS resources, such as an AWS WAF resource or an Amazon S3 bucket. The following sections provide details on how you can use AWS Identity and Access Management (IAM) and AWS WAF to help secure access to your resources.

## Authentication

You can access AWS as any of the following types of identities:

- **AWS account root user** – When you sign up for AWS, you provide an email address and password that is associated with your AWS account. This is your *AWS account root user*. Its credentials provide complete access to all of your AWS resources.

  **Important**
  For security reasons, we recommend that you use the root user only to create an *administrator*, which is an *IAM user* with full permissions to your AWS account. You can then use this administrator user to create other IAM users and roles with limited permissions. For more information, see IAM Best Practices and Creating an Admin User and Group in the *IAM User Guide*.
- **IAM user** – An IAM user is simply an identity within your AWS account that has specific custom permissions (for example, permissions to create a rule in AWS WAF). You can use an IAM user name and password to sign in to secure AWS webpages like the AWS Management Console, AWS Discussion Forums, or the AWS Support Center.

In addition to a user name and password, you can also generate access keys for each user. You can use these keys when you access AWS services programmatically, either through one of the several SDKs or by using the AWS Command Line Interface (CLI). The SDK and CLI tools use the access keys to cryptographically sign your request. If you don't use the AWS tools, you must sign the request yourself. AWS WAF supports *Signature Version 4*, a protocol for authenticating inbound API requests. For more information about authenticating requests, see Signature Version 4 Signing Process in the *AWS General Reference*.

- **IAM role** – An IAM role is another IAM identity that you can create in your account that has specific permissions. It is similar to an *IAM user*, but it is not associated with a specific person. An IAM role enables you to obtain temporary access keys that can be used to access AWS services and resources. IAM roles with temporary credentials are useful in the following situations:

  - **Federated user access** – Instead of creating an IAM user, you can use preexisting user identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an identity provider. For more information about federated users, see Federated Users and Roles in the *IAM User Guide*.

  - **Cross-account access** – You can use an IAM role in your account to grant another AWS account permissions to access your account's resources. For an example, see Tutorial: Delegate Access Across AWS Accounts Using IAM Roles in the *IAM User Guide*.

  - **AWS service access** – You can use an IAM role in your account to grant an AWS service permissions to access your account's resources. For example, you can create a role that allows Amazon Redshift to access an Amazon S3 bucket on your behalf and then load data from that bucket into an Amazon Redshift cluster. For more information, see Creating a Role to Delegate Permissions to an AWS Service in the *IAM User Guide*.

  - **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications running on an EC2 instance and making AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs running on the EC2 instance to get temporary credentials. For more information, see Using Roles for Applications on Amazon EC2 in the *IAM User Guide*.

# Access Control

You can have valid credentials to authenticate your requests, but unless you have permissions you can't create or access AWS WAF resources. For example, you must have permissions to create an AWS WAF *web ACL* or *rule*.

The following sections describe how to manage permissions for AWS WAF. We recommend that you read the overview first.

- Overview of Managing Access Permissions to Your AWS WAF Resources (p. 64)

# AWS Identity and Access Management

AWS WAF integrates with AWS Identity and Access Management (IAM), a service that lets your organization do the following:

- Create users and groups under your organization's AWS account
- Share your AWS account resources with users in the account
- Assign unique security credentials to each user
- Control user access to services and resources

For example, you can use IAM with AWS WAF to control which users in your AWS account can create a new web ACL.

For general information about IAM, see the following documentation:

- AWS Identity and Access Management (IAM)
- IAM Getting Started Guide
- IAM User Guide

# Overview of Managing Access Permissions to Your AWS WAF Resources

Every AWS resource is owned by an AWS account, and permissions to create or access a resource are governed by permissions policies. An account administrator can attach permissions policies to IAM identities (that is, users, groups, and roles), and some services also support attaching permissions policies to resources.

> **Note**
> An *account administrator* (or administrator user) is a user with administrator privileges. For more information, see IAM Best Practices in the *IAM User Guide*.

When granting permissions, you decide who is getting the permissions, the resources they get permissions for, and the specific operations that you want to allow on those resources.

## Topics

## AWS WAF Resources and Operations

In AWS WAF, the resources are *web ACLs* and *rules*. AWS WAF also supports conditions such as *byte match*, *IP match*, and *size constraint*.

These resources and conditions have unique Amazon Resource Names (ARNs) associated with them as shown in the following table.

| Name in WAF Console | Name in WAF SDK/CLI | ARN Format | |
|---|---|---|---|
| Web ACL | `WebACL` | arn:aws:waf::*account*:*webacl*/*ID* | |
| Rule | `Rule` | arn:aws:waf::*account*:*rule*/*ID* | |
| String match condition | `ByteMatchSet` | arn:aws:waf::*account*:*bytematchset*/*ID* | |
| SQL injection match condition | `SqlInjectionMatchSet` | arn:aws:waf::*account*:*sqlinjectionset*/*ID* | |
| Size constraint condition | `SizeConstraintSet` | arn:aws:waf::*account*:*sizeconstraintset*/*ID* | |
| IP match condition | `IPSet` | arn:aws:waf::*account*:*ipset*/*ID* | |
| Cross-site scripting match condition | `XssMatchSet` | arn:aws:waf::*account*:*xssmatchset*/*ID* | |

To allow or deny access to a subset of AWS WAF resources, include the ARN of the resource in the `resource` element of your policy. The ARNs for AWS WAF have the following format:

```
arn:aws:waf::account:resource/ID
```

Replace the *account*, *resource*, and *ID* variables with valid values. Valid values can be the following:

- *account*: The ID of your AWS account. You must specify a value.
- *resource*: The type of AWS WAF resource.
- *ID*: The ID of the AWS WAF resource, or a wildcard (*) to indicate all resources of the specified type that are associated with the specified AWS account.

For example, the following ARN specifies all web ACLs for the account `111122223333`:

```
arn:aws:waf::111122223333:webacl/*
```

For more information, see Resources in the *IAM User Guide*.

AWS WAF provides a set of operations to work with AWS WAF resources. For a list of available operations, see Actions.

## Understanding Resource Ownership

A *resource owner* is the AWS account that creates the resource. That is, the resource owner is the AWS account of the *principal entity* (the root account, an IAM user, or an IAM role) that authenticates the request that creates the resource. The following examples illustrate how this works:

- If you use the root account credentials of your AWS account to create an AWS WAF resource, your AWS account is the owner of the resource.

- If you create an IAM user in your AWS account and grant permissions to create an AWS WAF resource to that user, the user can create an AWS WAF resource. However, your AWS account, to which the user belongs, owns the AWS WAF resource.
- If you create an IAM role in your AWS account with permissions to create an AWS WAF resource, anyone who can assume the role can create an AWS WAF resource. Your AWS account, to which the role belongs, owns the AWS WAF resource.

## Managing Access to Resources

A *permissions policy* describes who has access to what. The following section explains the available options for creating permissions policies.

> **Note**
> This section discusses using IAM in the context of AWS WAF. It doesn't provide detailed information about the IAM service. For complete IAM documentation, see What Is IAM? in the *IAM User Guide*. For information about IAM policy syntax and descriptions, see AWS IAM Policy Reference in the *IAM User Guide*.

Policies that are attached to an IAM identity are known as *identity-based* policies and policies that are attached to a resource are known as *resource-based* policies. AWS WAF supports only identity-based policies.

### Topics

### Identity-Based Policies (IAM Policies)

You can attach policies to IAM identities. For example, you can do the following:

- **Attach a permissions policy to a user or a group in your account** – An account administrator can use a permissions policy that is associated with a particular user to grant permissions for that user to create an AWS WAF resource.
- **Attach a permissions policy to a role (grant cross-account permissions)** – You can attach an identity-based permissions policy to an IAM role to grant cross-account permissions. For example, the administrator in Account A can create a role to grant cross-account permissions to another AWS account (for example, Account B) or an AWS service as follows:
  1. Account A administrator creates an IAM role and attaches a permissions policy to the role that grants permissions on resources in Account A.
  2. Account A administrator attaches a trust policy to the role identifying Account B as the principal who can assume the role.
  3. Account B administrator can then delegate permissions to assume the role to any users in Account B. Doing this allows users in Account B to create or access resources in Account A. The principal in the trust policy also can be an AWS service principal if you want to grant an AWS service permissions to assume the role.

  For more information about using IAM to delegate permissions, see Access Management in the *IAM User Guide*.

The following is an example policy that grants permissions for the `waf:ListRules` action on all resources. In the current implementation, AWS WAF doesn't support identifying specific resources using the resource ARNs (also referred to as resource-level permissions) for some of the API actions, so you must specify a wildcard character (*):

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListRules",
            "Effect": "Allow",
            "Action": [
                "waf:ListRules"
            ],
            "Resource": "*"
        }
    ]
}
```

For more information about using identity-based policies with AWS WAF, see Using Identity-Based Policies (IAM Policies) for AWS WAF (p. 68). For more information about users, groups, roles, and permissions, see Identities (Users, Groups, and Roles) in the *IAM User Guide*.

### Resource-Based Policies

Other services, such as Amazon S3, also support resource-based permissions policies. For example, you can attach a policy to an S3 bucket to manage access permissions to that bucket. AWS WAF doesn't support resource-based policies.

## Specifying Policy Elements: Actions, Effects, Resources, and Principals

For each AWS WAF resource (see AWS WAF Resources and Operations (p. 64)), the service defines a set of API operations (see AWS WAF API Permissions: Actions, Resources, and Conditions Reference (p. 72)). To grant permissions for these API operations, AWS WAF defines a set of actions that you can specify in a policy. Note that performing an API operation can require permissions for more than one action. When granting permissions for specific actions, you also identify the resource on which the actions are allowed or denied.

The following are the most basic policy elements:

- **Resource** – In a policy, you use an Amazon Resource Name (ARN) to identify the resource to which the policy applies. For more information, see AWS WAF Resources and Operations (p. 64).
- **Action** – You use action keywords to identify resource operations that you want to allow or deny. For example, the `waf:CreateRule` permission allows the user permissions to perform the AWS WAF `CreateRule` operation.
- **Effect** – You specify the effect when the user requests the specific action. This can be either allow or deny. If you don't explicitly grant access to allow a resource, access is implicitly denied. You also can explicitly deny access to a resource, which you might do to make sure that a user cannot access it, even if a different policy grants access.
- **Principal** – In identity-based policies (IAM policies), the user that the policy is attached to is the implicit principal. AWS WAF doesn't support resource-based policies.

To learn more about IAM policy syntax and descriptions, see AWS IAM Policy Reference in the *IAM User Guide*.

For a table that shows all the AWS WAF API actions and the resources that they apply to, see AWS WAF API Permissions: Actions, Resources, and Conditions Reference (p. 72).

## Specifying Conditions in a Policy

When you grant permissions, you can use the IAM policy language to specify the conditions when a policy should take effect. For example, you might want a policy to be applied only after a specific date. For more information about specifying conditions in a policy language, see Condition in the *IAM User Guide*.

To express conditions, you use predefined condition keys. There are no condition keys specific to AWS WAF. However, there are AWS-wide condition keys that you can use as appropriate. For a complete list of AWS-wide keys, see Available Keys for Conditions in the *IAM User Guide*.

# Using Identity-Based Policies (IAM Policies) for AWS WAF

This topic provides examples of identity-based policies that demonstrate how an account administrator can attach permissions policies to IAM identities (that is, users, groups, and roles) and thereby grant permissions to perform operations on AWS WAF resources.

> **Important**
> We recommend that you first review the introductory topics that explain the basic concepts and options available for you to manage access to your AWS WAF resources. For more information, see Overview of Managing Access Permissions to Your AWS WAF Resources (p. 64).

The following shows an example of a permissions policy:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateFunctionPermissions",
            "Effect": "Allow",
            "Action": [
                "waf:ListWebACLs",
                "waf:ListRules",
                "waf:GetWebACL",
                "waf:GetRule",
                "cloudwatch:ListMetrics",
                "waf:GetSampledRequests"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PermissionToPassAnyRole",
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::account-id:role/*"
        }
    ]
}
```

The policy has two statements:

- The first statement grants permissions to view statistics for AWS WAF web ACLs, using the `waf:ListWebACLs`, `waf:ListRules`, `waf:GetWebACL`, `waf:GetRule`, `cloudwatch:ListMetrics`, and `waf:GetSampledRequests` actions. AWS WAF doesn't support permissions for some of these actions at the resource level. Therefore, the policy specifies a wildcard character (*) as the `Resource` value.
- The second statement grants permissions for the IAM action `iam:PassRole` on IAM roles. The wildcard character (*) at the end of the `Resource` value means that the statement allows permissions for the

iam:PassRole action on any IAM role. To limit these permissions to a specific role, replace the wildcard character (*) in the resource ARN with the specific role name.

The policy doesn't specify the Principal element because in an identity-based policy you don't specify the principal who gets the permissions. When you attach a policy to a user, the user is the implicit principal. When you attach a permissions policy to an IAM role, the principal identified in the role's trust policy gets the permissions.

For a table that shows all the AWS WAF API actions and the resources that they apply to, see AWS WAF API Permissions: Actions, Resources, and Conditions Reference (p. 72).

## Topics

## Permissions Required to Use the AWS WAF Console

The AWS WAF console provides an integrated environment for you to create and manage AWS WAF resources. The console provides many features and workflows that often require permissions to create an AWS WAF resource in addition to the API-specific permissions that are documented in the AWS WAF API Permissions: Actions, Resources, and Conditions Reference (p. 72). For more information about these additional console permissions, see Customer Managed Policy Examples (p. 69).

## AWS Managed (Predefined) Policies for AWS WAF

AWS addresses many common use cases by providing standalone IAM policies that are created and administered by AWS. Managed policies grant necessary permissions for common use cases so you can avoid having to investigate what permissions are needed. For more information, see AWS Managed Policies in the *IAM User Guide*.

The following AWS managed policies, which you can attach to users in your account, are specific to AWS WAF and are grouped by use case scenario:

- **AWSWAFReadOnlyAccess** – Grants read-only access to AWS WAF resources.
- **AWSWAFFullAccess** – Grants full access to AWS WAF resources.

> **Note**
> You can review these permissions policies by signing in to the IAM console and searching for specific policies there.

You also can create your own custom IAM policies to allow permissions for AWS WAF API operations and resources. You can attach these custom policies to the IAM users or groups that require those permissions or to custom execution roles (IAM roles) that you create for your AWS WAF resources.

## Customer Managed Policy Examples

The examples in this section provide a group of sample policies that you can attach to a user. If you are new to creating policies, we recommend that you first create an IAM user in your account and attach the policies to the user, in the sequence outlined in the steps in this section.

You can use the console to verify the effects of each policy as you attach the policy to the user. Initially, the user doesn't have permissions, and the user won't be able to do anything in the console. As you attach policies to the user, you can verify that the user can perform various operations in the console.

We recommend that you use two browser windows: one to create the user and grant permissions, and the other to sign in to the AWS Management Console using the user's credentials and verify permissions as you grant them to the user.

For examples that show how to create an IAM role that you can use as an execution role for your AWS WAF resource, see Creating IAM Roles in the *IAM User Guide*.

## Example Topics

- Example 1: Give Users Read-only Access to AWS WAF, CloudFront, and CloudWatch (p. 70)
- Example 2: Give Users Full Access to AWS WAF, CloudFront, and CloudWatch (p. 70)
- Example 3: Granting Access to a Specified AWS Account (p. 71)
- Example 4: Granting Access to a Specified Web ACL (p. 72)

## Create an IAM User

First, you need to create an IAM user, add the user to an IAM group with administrative permissions, and then grant administrative permissions to the IAM user that you created. You then can access AWS using a special URL and the user's credentials.

For instructions, see Creating Your First IAM User and Administrators Group in the *IAM User Guide*.

## Example 1: Give Users Read-only Access to AWS WAF, CloudFront, and CloudWatch

The following policy grants users read-only access to AWS WAF resources, to Amazon CloudFront web distributions, and to Amazon CloudWatch metrics. It's useful for users who need permission to view the settings in AWS WAF conditions, rules, and web ACLs to see which distribution is associated with a web ACL, and to monitor metrics and a sample of requests in CloudWatch. These users can't create, update, or delete AWS WAF resources:

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Action": [
            "waf:Get*",
            "waf:List*",
            "cloudfront:GetDistribution",
            "cloudfront:GetDistributionConfig",
            "cloudfront:ListDistributions",
            "cloudfront:ListDistributionsByWebACLId",
            "cloudwatch:ListMetrics",
            "cloudwatch:GetMetricStatistics"
         ],
         "Effect": "Allow",
         "Resource": "*"
      }
   ]
}
```

## Example 2: Give Users Full Access to AWS WAF, CloudFront, and CloudWatch

The following policy lets users perform any AWS WAF operation, perform any operation on CloudFront web distributions, and monitor metrics and a sample of requests in CloudWatch. It's useful for users who are AWS WAF administrators:

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Action": [
            "waf:*",
            "cloudfront:CreateDistribution",
            "cloudfront:GetDistribution",
            "cloudfront:GetDistributionConfig",
            "cloudfront:UpdateDistribution",
            "cloudfront:ListDistributions",
            "cloudfront:ListDistributionsByWebACLId",
            "cloudfront:DeleteDistribution",
            "cloudwatch:ListMetrics",
            "cloudwatch:GetMetricStatistics"
         ],
         "Effect": "Allow",
         "Resource": "*"
      }
   ]
}
```

We strongly recommend that you configure multi-factor authentication (MFA) for users who have administrative permissions. For more information, see Using Multi-Factor Authentication (MFA) Devices with AWS in the *IAM User Guide*.

## Example 3: Granting Access to a Specified AWS Account

This policy grants the following permissions to the account 444455556666:

- Full access to all AWS WAF operations and resources.
- Read and update access to all CloudFront distributions, which allows you to associate web ACLs and CloudFront distributions.
- Read access to all CloudWatch metrics and metric statistics, so that you can view CloudWatch data and a sample of requests in the AWS WAF console.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "waf:*"
         ],
         "Resource": [
            "arn:aws:waf::444455556666:*"
         ]
      },
      {
         "Effect": "Allow",
         "Action": [
            "cloudfront:GetDistribution",
            "cloudfront:GetDistributionConfig",
            "cloudfront:ListDistributions",
            "cloudfront:ListDistributionsByWebACLId",
            "cloudfront:UpdateDistributions",
            "cloudwatch:ListMetrics",
            "cloudwatch:GetMetricStatistics"
         ],
         "Resource": [
            "*"
         ]
      }
   ]
}
```

### Example 4: Granting Access to a Specified Web ACL

This policy grants the following permissions to the `webacl` ID 112233d7c-86b2-458b-af83-51c51example in the account 444455556666:

- Full access to AWS WAF `Get`, `Update`, and `Delete` operations and resources

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "waf:*"
         ],
         "Resource": [
            "arn:aws:waf::444455556666:webacl/112233d7c-86b2-458b-af83-51c51example"
         ]
      }
   ]
}
```

## AWS WAF API Permissions: Actions, Resources, and Conditions Reference

When you are setting up Access Control (p. 63) and writing permissions policies that you can attach to an IAM identity (identity-based policies), you can use the following table as a reference. The table lists each AWS WAF API operation, the corresponding actions for which you can grant permissions to perform the action, and the AWS resource for which you can grant the permissions. You specify the actions in the policy's `Action` field, and you specify the resource value in the policy's `Resource` field.

You can use AWS-wide condition keys in your AWS WAF policies to express conditions. For a complete list of AWS-wide keys, see Available Keys for Conditions in the *IAM User Guide*.

> **Note**
> To specify an action, use the `waf:` prefix followed by the API operation name (for example, `waf:CreateIPSet`).

**AWS WAF API and Required Permissions for Actions**

CreateByteMatchSet

**Action(s):** `waf:CreateByteMatchSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*:*`bytematchset`*/*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*:*`bytematchset`*/*`entity-ID`*

CreateIPSet

**Action(s):** `waf:CreateIPSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:ipset/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:ipset/entity-ID`

CreateRule

**Action(s):** `waf:CreateRule`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:rule/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:rule/entity-ID`

CreateRateBasedRule

**Action(s):** `waf:CreateRateBasedRule`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:rule/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:rule/entity-ID`

CreateSizeConstraintSet

**Action(s):** `waf:CreateSizeConstraintSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:sizeconstraintset/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:sizeconstraintset/entity-ID`

CreateSqlInjectionMatchSet

**Action(s):** `waf:CreateSqlInjectionMatchSet`,

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:sqlinjectionmatchset/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:sqlinjectionmatchset/entity-ID`

CreateWebACL

**Action(s):** `waf:CreateWebACL`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:webacl/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:webacl/entity-ID`

CreateXssMatchSet

**Action(s):** `waf:CreateXssMatchSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:xssmatchset/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:xssmatchset/entity-ID`

### DeleteByteMatchSet

**Action(s):** `waf:DeleteByteMatchSet`,

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:bytematchset/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:bytematchset/entity-ID`

### DeleteIPSet

**Action(s):** `waf:DeleteIPSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:ipset/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:ipset/entity-ID`

### DeleteRule

**Action(s):** `waf:DeleteRule`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:rule/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:rule/entity-ID`

### DeleteRateBasedRule

**Action(s):** `waf:DeleteRateBasedRule`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:rule/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:rule/entity-ID`

### DeleteSizeConstraintSet

**Action(s):** `waf:DeleteSizeConstraintSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:sizeconstraintset/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:sizeconstraintset/entity-ID`

### DeleteSqlInjectionMatchSet

**Action(s):** `waf:DeleteSqlInjectionMatchSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`sqlinjectionmatchset`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`sqlinjectionmatchset`*`/`*`entity-ID`*

### DeleteWebACL

**Action(s):** `waf:DeleteWebACL`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`webacl`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`webacl`*`/`*`entity-ID`*

### DeleteXssMatchSet

**Action(s):** `waf:DeleteXssMatchSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`xssmatchset`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`xssmatchset`*`/`*`entity-ID`*

### GetByteMatchSet

**Action(s):** `waf:GetByteMatchSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`bytematchset`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`bytematchset`*`/`*`entity-ID`*

### GetChangeToken

**Action(s):** `waf:GetChangeToken`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`changetoken`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`changetoken`*`/`*`entity-ID`*

### GetChangeTokenStatus

**Action(s):** `waf:GetChangeTokenStatus`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`changetoken`*`/`*`token-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`changetoken`*`/`*`token-ID`*

### GetIPSet

**Action(s):** `waf:GetIPSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`ipset`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`ipset`*`/`*`entity-ID`*

GetRule

**Action(s):** `waf:GetRule`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`rule`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`rule`*`/`*`entity-ID`*

GetRateBasedRule

**Action(s):** `waf:GetRateBasedRule`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`rule`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`rule`*`/`*`entity-ID`*

GetRateBasedRuleManagedKeys

**Action(s):** `waf:GetRateBasedRuleManagedKeys`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`rule`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`rule`*`/`*`entity-ID`*

GetSampledRequests

**Action(s):** `waf:GetSampledRequests`

**Resource:** Resource depends on the parameters that are specified in the API call. You must have access to the rule or webacl that correspond to the request for samples. For example:

Global (for Amazon CloudFront):arn:aws:waf::*`account-id`*:*`rule`*/example1 or arn:aws:waf::*`account-id`*:*`webacl`*/example2

Regional (for Application Load Balancer's):arn:aws:waf-regional:region:*`account-id`*:*`rule`*/example1 or arn:aws:waf-regional:region:*`account-id`*:*`webacl`*/example2

GetSizeConstraintSet

**Action(s):** `waf:GetSizeConstraintSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`sizeconstraintset`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`sizeconstraintset`*`/`*`entity-ID`*

GetSqlInjectionMatchSet

**Action(s):** `waf:GetSqlInjectionMatchSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`sqlinjectionmatchset`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`sqlinjectionmatchset`*`/`*`entity-ID`*

### GetWebACL

**Action(s):** `waf:GetWebACL`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`webacl`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`webacl`*`/`*`entity-ID`*

### GetXssMatchSet

**Action(s):** `waf:GetXssMatchSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`xssmatchset`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`xssmatchset`*`/`*`entity-ID`*

### ListByteMatchSets

**Action(s):** `waf:ListByteMatchSets`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`bytematchset`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`bytematchset`*`/`*`entity-ID`*

### ListIPSets

**Action(s):** `waf:ListIPSets`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`ipset`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`ipset`*`/`*`entity-ID`*

### ListRules

**Action(s):** `waf:ListRules`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`rule`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`rule`*`/`*`entity-ID`*

### ListRateBasedRules

**Action(s):** `waf:ListRateBasedRules`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:rule/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:rule/entity-ID`

### ListSizeConstraintSets

**Action(s):** `waf:ListSizeConstraintSets`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:sizeconstraintset/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:sizeconstraintset/entity-ID`

### ListSqlInjectionMatchSets

**Action(s):** `waf:ListSqlInjectionMatchSets`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:sqlinjectionmatchset/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:sqlinjectionmatchset/entity-ID`

### ListWebACLs

**Action(s):** `waf:ListWebACLs`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:webacl/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:webacl/entity-ID`

### ListXssMatchSets

**Action(s):** `waf:ListXssMatchSets`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:xssmatchset/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:xssmatchset/entity-ID`

### UpdateByteMatchSet

**Action(s):** `waf:UpdateByteMatchSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:bytematchset/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:bytematchset/entity-ID`

### UpdateIPSet

**Action(s):** `waf:UpdateIPSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`ipset`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`ipset`*`/`*`entity-ID`*

### UpdateRule

**Action(s):** `waf:UpdateRule`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`rule`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`rule`*`/`*`entity-ID`*

### UpdateRateBasedRule

**Action(s):** `waf:UpdateRateBasedRule`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`rule`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`rule`*`/`*`entity-ID`*

### UpdateSizeConstraintSet

**Action(s):** `waf:UpdateSizeConstraintSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`sizeconstraintset`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`sizeconstraintset`*`/`*`entity-ID`*

### UpdateSqlInjectionMatchSet

**Action(s):** `waf:UpdateSqlInjectionMatchSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`sqlinjectionmatchset`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`sqlinjectionmatchset`*`/`*`entity-ID`*

### UpdateWebACL

**Action(s):** `waf:UpdateWebACL`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::`*`account-id`*`:`*`webacl`*`/`*`entity-ID`*

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:`*`account-id`*`:`*`webacl`*`/`*`entity-ID`*

### UpdateXssMatchSet

**Action(s):** `waf:UpdateXssMatchSet`

**Resource:**

Global (for Amazon CloudFront): `arn:aws:waf::account-id:xssmatchset/entity-ID`

Regional (for Application Load Balancer's): `arn:aws:waf-regional:region:account-id:xssmatchset/entity-ID`

# AWS WAF Limits

AWS WAF has default limits on the number of entities per account. You can request an increase in these limits.

| **Resource** | **Default Limit** |
|---|---|
| Web ACLs per AWS account | 50 |
| Rules per AWS account | 100 |
| Conditions per AWS account | 100 of each condition type (For example: 100 size constraint conditions, 100 IP match conditions, and so on) |
| Requests per Second | 10,000 per web ACL* |

*This limit applies only to AWS WAF on an Application Load Balancer. Requests per Second (RPS) limits for AWS WAF on CloudFront are the same as the RPS limits support by CloudFront that is described in the CloudFront Developer Guide.

The following limits on AWS WAF entities can't be changed.

| **Resource** | **Limit** |
|---|---|
| Rules per web ACL | 10 |
| Conditions per rule | 10 |
| IP address ranges (in CIDR notation) per IP match condition | 10,000 |
| IP addresses blocked per rate-based rule | 10,000 |
| Minimum rate-based rule rate limit per 5 minute period | 2000 |
| Filters per cross-site scripting match condition | 10 |
| Filters per size constraint condition | 10 |
| Filters per SQL injection match condition | 10 |
| Filters per string match condition | 10 |
| In string match conditions, the number of characters in HTTP header names, when you've configured AWS WAF to inspect the headers in web requests for a specified value | 40 |

| Resource | Limit |
|---|---|
| In string match conditions, the number of bytes in the value that you want AWS WAF to search for | 50 |

# AWS Shield

AWS provides AWS Shield Standard and AWS Shield Advanced for protection against DDoS attacks. AWS Shield Standard is automatically included at no extra cost beyond what you already pay for AWS WAF and your other AWS services. For added protection against DDoS attacks, AWS offers AWS Shield Advanced. AWS Shield Advanced provides expanded DDoS attack protection for your Elastic Load Balancing load balancers, CloudFront distributions, and Amazon Route 53 hosted zones.

Topics

## How AWS Shield Works

A distributed denial of service (DDoS) attack is an attack in which multiple compromised systems attempt to flood a target, such as a network or web application, with traffic. A DDoS attack can prevent legitimate users from accessing a service and can cause the system to crash due to the overwhelming traffic volume.

AWS provides two levels of protection against DDoS attacks: AWS Shield Standard and AWS Shield Advanced.

### AWS Shield Standard

AWS services and technologies are built from the ground up to provide resilience in the face of network and transport layer DDoS attacks. For web application attacks, you also can use AWS WAF to configure web access control lists (web ACLs) that target network layer DDoS request patterns and help to minimize the effects of a DDoS attack. This DDoS protection, known as AWS Shield Standard, is included with AWS WAF. AWS Shield Standard is provided at no additional cost beyond what you are already paying for AWS WAF and your other AWS services.

### AWS Shield Advanced

AWS Shield Advanced provides expanded DDoS attack protection for your Elastic Load Balancing load balancers, CloudFront distributions, and Amazon Route 53 hosted zones.

AWS Shield Advanced includes intelligent DDoS attack detection and mitigation for not only for network layer (layer 3) and transport layer (layer 4) attacks, but also for application layer (layer 7) attacks. As an AWS Shield Advanced customer, you can contact a 24x7 DDoS response team (DRT) for assistance during a DDoS attack. You also have exclusive access to advanced, real-time metrics and reports for extensive visibility into attacks on your AWS resources.

AWS Shield Advanced also offers some cost protection against spikes in your AWS bill that could result from a DDoS attack. This cost protection is provided for your Elastic Load Balancing load balancers, CloudFront distributions, and Amazon Route 53 hosted zones.

AWS WAF is included with AWS Shield Advanced at no extra cost. For more information about AWS Shield Advanced pricing, see AWS Shield Advanced Pricing.

# Types of DDoS Attacks

AWS Shield Advanced provides expanded protection against many types of attacks. For example:

**User Datagram Protocol (UDP) reflection attacks**

An attacker can spoof the source of a request and use UDP to elicit a large response from the server. The extra network traffic directed towards the spoofed, attacked IP address can slow the targeted server and prevent legitimate users from accessing needed resources.

**SYN flood**

The intent of an SYN flood attack is to exhaust the available resources of a system by leaving connections in a half-open state. When a user connects to a TCP service like a web server, the client sends a SYN packet. The server returns an acknowledgment, and the client returns its own acknowledgement, completing the three-way handshake. In an SYN flood, the third acknowledgment is never returned, and the server is left waiting for a response. This can prevent other users from connecting to the server.

**DNS query flood**

In a DNS query flood, an attacker uses multiple DNS queries to exhaust the resources of a DNS server. AWS Shield Advanced can help provide protection against DNS query flood attacks on Amazon Route 53 DNS servers.

**HTTP flood/cache-busting (layer 7) attacks**

With an HTTP flood, including GET and POST floods, an attacker sends multiple HTTP requests that appear to be from a real user of the web application. Cache-busting attacks are a type of HTTP flood that uses variations in the HTTP request's query string that prevent use of edge-located cached content and forces the content to be served from the origin web server, causing additional and potentially damaging strain on the origin web server.

# About the AWS DDoS Response Team (DRT)

With AWS Shield Advanced, complex DDoS events can be escalated to the AWS DDoS Response team (DRT), which has deep experience in protecting AWS, Amazon.com, and its subsidiaries.

For layer 3 and layer 4 attacks, AWS provides automatic attack detection and proactively applies mitigations on your behalf. For layer 7 DDoS attacks, AWS attempts to detect and notify AWS Shield Advanced customers through CloudWatch alarms, but does not apply mitigations proactively. This is to avoid inadvertently dropping valid user traffic.

AWS Shield Advanced customers have two options to mitigate layer 7 attacks:

- **Provide your own mitigations:** AWS WAF is included with AWS Shield Advanced at no extra cost. You can create your own AWS WAF rules to mitigate the DDoS attacks. AWS provides preconfigured templates to get you started quickly. The templates include a set of AWS WAF rules that are designed to block common web-based attacks. You can customize the templates to fit your business needs. For more information, see AWS WAF Security Automations.

  In this case, the DRT is not involved. You can, however, engage the DRT for guidance on implementing best practices such as AWS WAF common protections.
- **Engage the DRT:** If you want additional support in addressing an attack, you can contact the AWS Support Center. Critical and urgent cases are routed directly to DDoS experts. With AWS Shield Advanced, complex cases can be escalated to the DRT, which has deep experience in protecting AWS, Amazon.com, and its subsidiaries. If you are an AWS Shield Advanced customer, you also can request special handling instructions for high severity cases.

  The response time for your case depends on the severity that you select and the response times, which are documented on the AWS Support Plans page.

  The DRT helps you triage the DDoS attack to identify attack signatures and patterns. With your consent, the DRT creates and deploys AWS WAF rules to mitigate the attack.

When AWS Shield Advanced detects a large layer 7 attack against one of your applications, the DRT might proactively contact you. The DRT triages the DDoS incident and creates AWS WAF mitigations. The DRT then contacts you for consent to apply the AWS WAF rules.

> **Important**
> The DRT can help you to analyze suspicious activity and assist you to mitigate the issue. This mitigation often requires the DRT to create or update web access control lists (web ACLs) in your account. However, they need your permission to do so. We recommend that as part of enabling AWS Shield Advanced, you follow the steps in Step 3: Authorize the DDoS Response Team to Create Rules and Web ACLs on Your Behalf (p. 89) to proactively provide the DRT with the needed permissions. Providing permission ahead of time helps prevent any delays in the event of an actual attack.

# Help Me Choose a Protection Plan

In many cases, AWS Shield Standard protection is sufficient for your needs. AWS services and technologies are built to provide resilience in the face of the most common DDoS attacks. Supplementing this built-in protection with AWS WAF and a combination of other AWS services as a defense-in-depth strategy typically provides adequate attack protection and mitigation. Further, if you have the technical expertise and want full control over monitoring for and mitigating layer 7 attacks, AWS Shield Standard is likely the appropriate choice. For additional resources to help you design your own DDoS protection, see our Tutorials (p. 17).

If your business or industry is a likely target of DDoS attacks, or if you prefer to let AWS handle the majority of DDoS protection and mitigation responsibilities for layer 3, layer 4, and layer 7 attacks, AWS Shield Advanced might be the best choice. AWS Shield Advanced not only provides layer 3 and layer 4 protection and mitigation, but also includes AWS WAF at no extra charge and DRT assistance for layer 7 attacks. If you use AWS WAF and AWS Shield Standard, you must design your own layer 7 protection and mitigation processes.

AWS Shield Advanced customers also benefit from detailed information about DDoS attacks against their AWS resources. While AWS Shield Standard provides automatic protection for the most common layer 3 and layer 4 attacks, visibility into the details of those attacks is limited. AWS Shield Advanced provides you with extensive data about the details of both layer 3, layer 4, and layer 7 DDoS attacks.

AWS Shield Advanced also offers cost protection for DDoS attacks against your AWS resources. This valuable feature helps prevent unexpected spikes in your bill caused by DDoS attacks. If cost predictability is important to you, AWS Shield Advanced can offer that stability.

The following table shows a comparison of AWS Shield Standard and AWS Shield Advanced.

| Feature | AWS Shield Standard | AWS Shield Advanced |
|---|---|---|
| **Active Monitoring** | | |
| Network flow monitoring | Yes | Yes |
| Automatic always-on detection | Yes | Yes |
| Automated application (layer 7) traffic monitoring | | Yes |
| **DDoS Mitigations** | | |
| Helps protect against common DDoS attacks, such as SYN flood and UDP reflection attacks | Yes | Yes |
| Access to additional DDoS mitigation capacity | | Yes |
| Custom application layer (layer 7) mitigations | Yes, through user-created AWS WAF ACLs. Incurs standard AWS WAF charges. | Yes, through user-created or DRT-created AWS WAF ACLs. Included as part of the AWS Shield Advanced subscription. |
| Instant rule updates | Yes, through user-created AWS WAF ACLs. Incurs standard AWS WAF charges. | Yes |
| AWS WAF for app vulnerability protection | Yes, through user-created AWS WAF ACLs. Incurs standard AWS WAF charges. | Yes |
| **Visibility and Reporting** | | |
| Layer 3/4 attack notification | | Yes |
| Layer 3/4 attack forensics reports (source IP, attack vector, and more) | | Yes |

| Feature | AWS Shield Standard | AWS Shield Advanced |
|---|---|---|
| Layer 7 attack notification | Yes, through AWS WAF. Incurs standard AWS WAF charges. | Yes |
| Layer 7 attack forensics reports (Top talkers report, sampled requests, and more) | Yes, through AWS WAF. Incurs standard AWS WAF charges. | Yes |
| Layer 3/4/7 attack historical report | | Yes |
| **DDoS Response Team Support** | | |
| Incident management during high severity events | | Yes |
| Custom mitigations during attacks | | Yes |
| Post-attack analysis | | Yes |
| **Cost Protection** | | |
| Reimburse Amazon Route 53 DNS DDoS charges | | Yes |
| Reimburse CloudFront DDoS charges | | Yes |
| Reimburse Elastic Load Balancing (ELB) DDoS charges | | Yes |

**Note**
Although both AWS Shield Standard and AWS Shield Advanced provide significant protection against DDoS attacks, we recommend that you also use Amazon CloudWatch and AWS CloudTrail to monitor all of your AWS services. For information about monitoring AWS WAF by using CloudWatch and CloudTrail, see Monitoring AWS WAF and AWS Shield Advanced (p. 91) and Logging AWS WAF API Calls with AWS CloudTrail (p. 95).

# AWS Shield Advanced Pricing

## AWS Shield Advanced and AWS Shield Standard Pricing

AWS Shield Standard is included with your AWS services at no additional cost.

AWS Shield Advanced pricing is detailed on the AWS Shield Advanced Pricing page. AWS Shield Advanced does have an additional cost, but AWS Shield Advanced customers do not pay for AWS WAF separately; it is included as part of the AWS Shield Advanced service. Further, AWS Shield Advanced charges do not increase with attack volume. This provides a predictable cost for the extended protection.

The AWS Shield Advanced fee applies for each business that is subscribed to AWS Shield Advanced. If your business has multiple AWS accounts, you pay just one Shield Advanced monthly fee as long as all the AWS accounts are in the same Consolidated Billing account family. Further, you must own all the AWS accounts and resources in the account.

# Getting Started with AWS Shield Advanced

This topic shows you how to get started with AWS Shield Advanced. For best results, perform all four of the following steps in sequence.

Topics

## Step 1: Enable and Configure AWS Shield Advanced

AWS Shield Advanced provides advanced DDoS detection and mitigation protection for network layer (layer 3), transport layer (layer 4), and application layer (layer 7) attacks.

> **Important**
> You must enable Shield Advanced for each AWS account that you want to protect. For more information, see Enabling and Configuring AWS Shield Advanced for Multiple Accounts.

**To enable and configure AWS Shield Advanced**

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. If this is your first time logging in to the AWS WAF console, choose **Get started with AWS Shield Advanced**. Otherwise, choose **Protected resources**.
3. Choose **Activate AWS Shield Advanced**.
4. Choose the resource type and resource to protect.
5. For **Name**, enter a friendly name to help you identify the AWS resources that are protected. For example, `My CloudFront AWS Shield Advanced distributions`.
6. (Optional) For **Web DDoS attack**, select **Enable**. You are prompted to associate an existing web ACL with these resources, or create a web ACL if you don't have one yet.

You can disable this protection later by following the steps described in Removing AWS Shield Advanced from an AWS Resource (p. 90).

7. Choose **Add DDoS protection**.

To protect additional resources, see Step 2: Add AWS Shield Advanced Protection to AWS Resources (p. 88).

## Enabling and Configuring AWS Shield Advanced for Multiple Accounts

You must enable Shield Advanced for each AWS account that you want to protect. To do so, follow the procedure in Enable and Configure Shield Advanced for each account, each time logging in with a different account.

The first time that you enable Shield Advanced from an account, you are presented with a pricing agreement. The AWS Shield Advanced fee applies for each business that is subscribed to AWS Shield Advanced. So although the pricing agreement displays in the console each time that you enable Shield Advanced from a new account, if your business has multiple AWS accounts, you pay just one Shield Advanced monthly fee as long as all the AWS accounts are in the same Consolidated Billing account family. Further, you must own all the AWS accounts and resources in the account.

# Step 2: Add AWS Shield Advanced Protection to AWS Resources

As part of enabling Shield Advanced for an account, you choose an initial resource to protect. You likely will want to add protection to more resources. Shield Advanced offers advanced monitoring and protection for up to 100 resources that include any combination of CloudFront distributions, Amazon Route 53 hosted zones, or Elastic Load Balancing resources. If you want to increase these limits, contact the AWS Support Center.

> **Important**
> You must complete Step 1: Enable and Configure AWS Shield Advanced (p. 87) before you start Step 2.

**To add protection for an AWS resource**

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. Choose **Protected resources**.
3. Choose **Add DDoS protection**.
4. Choose the resource type and resource to protect. For Classic Load Balancer and Application Load Balancer resources, you also must choose a region.
5. For **Name**, type a friendly name to help you identify the AWS resources that are protected. For example, `My CloudFront AWS Shield Advanced distributions`.
6. (Optional) For **Web DDoS attack**, select **Enable**. You are prompted to associate an existing web ACL with these resources, or create a web ACL if you don't have one yet.

   You can disable this protection later by following the steps described in Removing AWS Shield Advanced from an AWS Resource (p. 90).
7. Choose **Add DDoS protection**.

After you have added DDoS protection to all the appropriate resources, go to Step 3: Authorize the DDoS Response Team to Create Rules and Web ACLs on Your Behalf (p. 89).

# Step 3: Authorize the DDoS Response Team to Create Rules and Web ACLs on Your Behalf

One of the benefits of AWS Shield Advanced is support from the DDoS response team (DRT). When you experience a potential DDoS attack, you can contact the AWS Support Center. If necessary, the Support Center escalates your issue to the DRT. The DRT can help you analyze the suspicious activity and assist you in mitigating the issue. This mitigation often involves creating or updating AWS WAF rules and web access control lists (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, but the team needs your authorization to do so. We recommend that as part of enabling AWS Shield Advanced, you proactively provide the DRT with the needed authorization. Providing authorization ahead of time helps prevent mitigation delays in the event of an actual attack.

To authorize the DRT to inspect your AWS WAF configuration and create and update AWS WAF rules and web ACLs on your behalf, you must use an AWS CloudFormation template that creates a cross-account IAM role. This role establishes a trust relationship with the DRT support account and applies a policy giving the DRT full access to your AWS WAF resources. The policy enables the DRT to inspect your AWS WAF configuration and create or update AWS WAF rules and web ACLs.

The policy gives the DRT access only to your AWS WAF and Shield resources. By running this template, you authorize the DRT to inspect your AWS WAF and Shield configuration and create and update AWS WAF rules and web ACLs on your behalf. The DRT takes these actions only if explicitly authorized by you, as described on this page.

> **Important**
> You must complete Step 1: Enable and Configure AWS Shield Advanced (p. 87) and Step 2: Add AWS Shield Advanced Protection to AWS Resources (p. 88) before you start this procedure.

**To authorize the DRT to mitigate potential attacks on your behalf**

1. Use this link to create an AWS CloudFormation stack: AWS CloudFormation DRT stack. The link pre-populates the AWS CloudFormation **Select Template** page with the correct AWS CloudFormation template name.
2. Choose **Next**.
3. For **Stack name**, type a name, and then choose **Next**.
4. On the **Options** page, keep the defaults, and then choose **Next**.
5. Review the summary, and then choose **Create**.

After you authorize the DRT to act on your behalf, you should deploy AWS WAF security automations as described in Step 4: Deploy AWS WAF Security Automations (p. 89).

# Step 4: Deploy AWS WAF Security Automations

AWS provides preconfigured templates that include a set of AWS WAF rules, which you can customize to best fit your needs. These templates are designed to block common web-based attacks such as bad bots, SQL injection, cross-site scripting (XSS), HTTP floods, and known-attacker attacks. In addition to enabling Shield Advanced and specifying resources for Shield Advanced protection, you also should use these preconfigured templates.

For more information, see AWS WAF Security Automations. AWS WAF is included with Shield Advanced at no additional cost.

# Removing AWS Shield Advanced from an AWS Resource

You can remove AWS Shield Advanced protection from any of your resources at any time.

**Remove AWS Shield Advanced protection from an AWS resource**

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. Choose **Protected resources**.
3. Choose the radio button next to the resource.
4. Choose **Delete protection**.

These steps remove AWS Shield Advanced protection from a specific resource. They do not cancel your AWS Shield Advanced subscription. You will continue to be charged for the service. For more information about your AWS Shield Advanced subscription, contact the AWS Support Center.

# AWS Shield Advanced: Requesting a Credit

If you are subscribed to AWS Shield Advanced and a DDoS attack results in additional charges for your Amazon CloudFront, Elastic Load Balancing, or Amazon Route 53 services, you can apply for a credit for the charges by submitting a billing case through the AWS Support Center.

If the AWS Shield Advanced team determines that the incident is a valid DDoS attack and that the underlying services scaled to absorb the attack, AWS provides account credit for charges incurred due to the attack. For example, if your legitimate CloudFront data transfer usage during the attack period was 20 GB, but due to the attack you incurred charges for 200 GB of incremental data transfer, AWS provides credit to offset the incremental data transfer charges. AWS automatically applies all credits toward your future monthly bills. Credits are applied to the specific AWS service affected by the DDoS attack and cannot be used for payment for other AWS services. Credits are valid for 12 months.

> **Important**
> To be eligible for a credit, AWS must receive your credit request by the end of the second billing cycle after the incident occurred.

To request your credit, submit a billing query to the AWS Support Center that contains the following information:

- The words “DDoS Concession” in the subject line
- The dates and times of each incident interruption that you are claiming
- The AWS services (Amazon CloudFront, Elastic Load Balancing, or Amazon Route 53) and specific resources that were affected by the DDoS activity

# AWS Shield Advanced Limits

AWS Shield Advanced offers advanced monitoring and protection for up to 100 CloudFront distributions, Amazon Route 53 hosted zones, or Elastic Load Balancing load balancers combined. If you want to increase these limits, contact the AWS Support Center.

# Monitoring AWS WAF and AWS Shield Advanced

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS WAF and for identifying possible DDoS attacks using AWS Shield. As you start monitoring AWS WAF and AWS Shield, you should create a monitoring plan that includes answers to the following questions:

- What are your monitoring goals?
- What resources will you monitor?
- How often will you monitor these resources?
- What monitoring tools will you use?
- Who will perform the monitoring tasks?
- Who should be notified when something goes wrong?

The next step is to establish a baseline for normal performance in your environment, by measuring performance at various times and under different load conditions. As you monitor AWS WAF and related services, store historical monitoring data so that you can compare it with current performance data, identify normal performance patterns and performance anomalies, and devise methods to address issues.

For AWS WAF, you should monitor the following items at a minimum to establish a baseline:

- The number of allowed web requests
- The number of blocked web requests

Topics

# Monitoring Tools

AWS provides various tools that you can use to monitor AWS WAF and AWS Shield. You can configure some of these tools to do the monitoring for you, while some of the tools require manual intervention. We recommend that you automate monitoring tasks as much as possible.

## Automated Monitoring Tools

You can use the following automated monitoring tools to watch AWS WAF and AWS Shield Advanced and report when something is wrong:

- **Amazon CloudWatch Alarms** – Watch a single metric over a time period that you specify, and perform one or more actions based on the value of the metric relative to a given threshold over a number of time periods. The action is a notification sent to an Amazon Simple Notification Service (Amazon SNS) topic or Auto Scaling policy. CloudWatch alarms do not invoke actions simply because they are in a particular state; the state must have changed and been maintained for a specified number of periods. For more information, see Monitoring with Amazon CloudWatch (p. 93).

  Not only can you use CloudWatch to monitor AWS WAF and Shield Advanced metrics as described in Monitoring with Amazon CloudWatch (p. 93), you also should use CloudWatch to monitor activity for you Elastic Load Balancing resources and Amazon CloudFront distributions. For more information, see CloudWatch Metrics for Your Application Load Balancer and Monitoring CloudFront Activity Using CloudWatch.
- **Amazon CloudWatch Logs** – Monitor, store, and access your log files from AWS CloudTrail or other sources. For more information, see Monitoring Log Files in the *Amazon CloudWatch User Guide*.
- **Amazon CloudWatch Events** – Match events and route them to one or more target functions or streams to make changes, capture state information, and take corrective action. For more information, see Using Events in the *Amazon CloudWatch User Guide*.
- **AWS CloudTrail Log Monitoring** – Share log files between accounts, monitor CloudTrail log files in real time by sending them to CloudWatch Logs, write log-processing applications in Java, and validate that your log files have not changed after delivery by CloudTrail. For more information, see Logging AWS WAF API Calls with AWS CloudTrail (p. 95) and Working with CloudTrail Log Files in the *AWS CloudTrail User Guide*.

## Manual Monitoring Tools

Another important part of monitoring AWS WAF and AWS Shield involves manually monitoring those items that the CloudWatch alarms don't cover. You can view the AWS WAF, AWS Shield Advanced, CloudWatch, and other AWS console dashboards to see the state of your AWS environment. We recommend that you also check the log files for your web ACLs and rules.

- View the AWS WAF dashboard:
  - On the **Requests** tab of the AWS WAF **Web ACLs** page, view a graph of total requests and requests that match each rule that you have created. For more information, see Viewing a Sample of the Web Requests That CloudFront or an Application Load Balancer Has Forwarded to AWS WAF (p. 59).
- View the CloudWatch home page for the following:
  - Current alarms and status
  - Graphs of alarms and resources
  - Service health status

  In addition, you can use CloudWatch to do the following:
  - Create customized dashboards to monitor the services that you care about

- Graph metric data to troubleshoot issues and discover trends
- Search and browse all of your AWS resource metrics
- Create and edit alarms to be notified of problems

# Monitoring with Amazon CloudWatch

You can monitor web requests and web ACLs and rules using CloudWatch, which collects and processes raw data from AWS WAF into readable, near real-time metrics. These statistics are recorded for a period of two weeks, so that you can access historical information and gain a better perspective on how your web application or service is performing. For more information, see What is CloudWatch in the *Amazon CloudWatch User Guide*.

## Creating CloudWatch Alarms

You can create a CloudWatch alarm that sends an Amazon SNS message when the alarm changes state. An alarm watches a single metric over a time period that you specify, and performs one or more actions based on the value of the metric relative to a specified threshold over a number of time periods. The action is a notification sent to an Amazon SNS topic or Auto Scaling policy. Alarms invoke actions for sustained state changes only. CloudWatch alarms do not invoke actions simply because they are in a particular state; the state must have changed and been maintained for a specified number of periods.

## AWS WAF and AWS Shield Advanced Metrics and Dimensions

You can use the following procedures to view the metrics for AWS WAF and Shield Advanced.

**To view metrics using the CloudWatch console**

Metrics are grouped first by the service namespace, and then by the various dimension combinations within each namespace.

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. If necessary, change the region. From the navigation bar, select the region where your AWS resources are located. For more information, see Regions and Endpoints.
3. In the navigation pane, choose **Metrics**.
4. On the **All metrics** tab, choose the appropriate service.

**To view metrics using the AWS CLI**

- For AWS WAF, at a command prompt use the following command:

  ```
  aws cloudwatch list-metrics --namespace "AWS/WAF"
  ```

  For Shield Advanced, at a command prompt use the following command:

  ```
  aws cloudwatch list-metrics --namespace "DDoSProtection"
  ```

# AWS WAF Metrics

The `WAF` namespace includes the following metrics.

| Metric | Description |
|---|---|
| `AllowedRequests` | The number of allowed web requests.<br>Units: Count<br>Dimensions: `Rule`, `WebACL`<br>Valid statistics: Sum |
| `BlockedRequests` | The number of blocked web requests.<br>Units: Count<br>Dimensions: `Rule`, `WebACL`<br>Valid statistics: Sum |
| `CountedRequests` | The number of counted web requests.<br>A counted web request is one that matches all of the conditions in a particular rule. Counted web requests are typically used for testing.<br>Units: Count<br>Dimensions: `Rule`, `WebACL`<br>Valid statistics: Sum |

# AWS WAF Dimensions

| Dimension | Description |
|---|---|
| `Rule` | The name of the rule, or one of the following:<br>• ALL, which represents the set of all rules.<br>• Default_Action, which represents the action assigned to any request that does not match any rule with either an allow or block action. |
| `WebACL` | The name of the web ACL. |

# Shield Advanced Metrics

## AWS Shield Advanced Metrics

AWS Shield Advanced includes the following metrics.

| Metric | Description |
|---|---|
| RequestCount | The number of requests during an attack.<br>Reporting criteria: Non-zero value during an attack. Zero when there is no attack.<br>Statistics: The most useful statistic is Sum.<br>Units: Count |
| ByteCount | The number of bytes per second during an attack.<br>Reporting criteria: Non-zero value during an attack. Zero when there is no attack.<br>Statistics: The most useful statistic is Sum.<br>Units: Bytes |

AWS Shield Advanced reports metrics to CloudWatch only when an attack on an AWS resource is detected. If there are no attacks for a specified period, AWS Shield Advanced reports zero.

### Creating AWS Shield Advanced Alarms

You can use these Shield Advanced metrics for CloudWatch alarms. CloudWatch alarms send notifications or automatically make changes to the resources you are monitoring based on rules that you define.

Detailed instructions for creating a CloudWatch alarm are in the Amazon CloudWatch User Guide. When creating the alarm in the CloudWatch console, after choosing **Create an alarm**, choose **AWSDDOSProtection Metrics** to use these Shield Advanced metrics. You can then create an alarm based on a specific volume of traffic, or you can trigger the alarm whenever either of the above metrics is greater than zero. Because Shield Advanced metrics are only reported when an attack is detected, the second option would trigger an alarm for any potential attack observed by Shield Advanced.

> **Note**
> The **AWSDDOSProtection Metrics** are only available to Shield Advanced customers.

For more information, see What is CloudWatch in the *Amazon CloudWatch User Guide*.

# Logging AWS WAF API Calls with AWS CloudTrail

AWS WAF is integrated with CloudTrail, a service that captures all the AWS WAF API calls and delivers the log files to an Amazon S3 bucket that you specify. CloudTrail captures API calls from the AWS WAF console or from your code to the AWS WAF API. Using the information collected by CloudTrail, you can identify the request that was made to AWS WAF, the source IP address from which the request was made, who made the request, when it was made, and so on.

To learn more about CloudTrail, including how to configure and enable it, see the AWS CloudTrail User Guide.

## AWS WAF Information in CloudTrail

When CloudTrail logging is enabled in your AWS account, API calls made to AWS WAF actions are tracked in CloudTrail log files, where they are written with other AWS service records. CloudTrail determines when to create and write to a new file based on a time period and file size.

All AWS WAF actions are logged by CloudTrail and are documented in the AWS WAF API Reference. For example, calls to `ListWebACL`, `UpdateWebACL`, and `DeleteWebACL` generate entries in the CloudTrail log files.

Every log entry contains information about who generated the request. The user identity information in the log entry helps you determine the following:

- Whether the request was made with root or IAM user credentials
- Whether the request was made with temporary security credentials for a role or federated user
- Whether the request was made by another AWS service

For more information, see CloudTrail userIdentity Element.

You can store your log files in your Amazon S3 bucket for as long as you want, but you also can define Amazon S3 lifecycle rules to archive or delete log files automatically. By default, your log files are encrypted with Amazon S3 server-side encryption (SSE).

If you want to be notified upon log file delivery, you can configure CloudTrail to publish Amazon SNS notifications when new log files are delivered. For more information, see Configuring Amazon SNS Notifications for CloudTrail.

You also can aggregate AWS WAF log files from multiple AWS Regions and multiple AWS accounts into a single Amazon S3 bucket.

For more information, see Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts.

# Understanding AWS WAF Log File Entries

CloudTrail log files can contain one or more log entries. Each entry lists multiple JSON-formatted events. A log entry represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. Log entries are not an ordered stack trace of the public API calls, so they do not appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the following actions (see the `eventName` elements):

- `CreateRule`
- `GetRule`
- `UpdateRule`
- `DeleteRule`

```
{
  "Records": [
   {
      "eventVersion": "1.03",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAIEP4IT4TPDEXAMPLE",
        "arn": "arn:aws:iam::777777777777:user/nate",
        "accountId": "777777777777",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "nate"
      },
      "eventTime": "2016-04-25T21:35:14Z",
      "eventSource": "waf.amazonaws.com",
      "eventName": "CreateRule",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "AWS Internal",
      "userAgent": "console.amazonaws.com",
      "requestParameters": {
        "name": "0923ab32-7229-49f0-a0e3-66c81example",
        "changeToken": "l9434322-8685-4ed2-9c5b-9410bexample",
        "metricName": "0923ab32722949f0a0e366c81example"
      },
      "responseElements": {
        "rule": {
          "metricName": "0923ab32722949f0a0e366c81example",
          "ruleId": "12132e64-6750-4725-b714-e7544example",
          "predicates": [

          ],
          "name": "0923ab32-7229-49f0-a0e3-66c81example"
        },
        "changeToken": "l9434322-8685-4ed2-9c5b-9410bexample"
      },
      "requestID": "4e6b66f9-d548-11e3-a8a9-73e33example",
      "eventID": "923f4321-d378-4619-9b72-4605bexample",
      "eventType": "AwsApiCall",
      "apiVersion": "2015-08-24",
      "recipientAccountId": "777777777777"
    },
    {
      "eventVersion": "1.03",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAIEP4IT4TPDEXAMPLE",
        "arn": "arn:aws:iam::777777777777:user/nate",
        "accountId": "777777777777",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "nate"
      },
      "eventTime": "2016-04-25T21:35:22Z",
      "eventSource": "waf.amazonaws.com",
      "eventName": "GetRule",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "AWS Internal",
      "userAgent": "console.amazonaws.com",
      "requestParameters": {
        "ruleId": "723c2943-82dc-4bc1-a29b-c7d73example"
      },
      "responseElements": null,
      "requestID": "8e4f3211"-d548-11e3-a8a9-73e33example",
      "eventID": "an236542-d1f9-4639-bb3d-8d2bbexample",
      "eventType": "AwsApiCall",
      "apiVersion": "2015-08-24",
      "recipientAccountId": "777777777777"
    },
    {
      "eventVersion": "1.03",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAIEP4IT4TPDEXAMPLE",
        "arn": "arn:aws:iam::777777777777:user/nate",
        "accountId": "777777777777",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "nate"
      },
      "eventTime": "2016-04-25T21:35:13Z",
      "eventSource": "waf.amazonaws.com",
      "eventName": "UpdateRule",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "AWS Internal",
      "userAgent": "console.amazonaws.com",
      "requestParameters": {
        "ruleId": "7237b123-7903-4d9e-8176-9d71dexample",
        "changeToken": "32343a11-35e2-4dab-81d8-6d408example",
        "updates": [
          {
            "predicate": {
              "type": "SizeConstraint",
              "dataId": "9239c032-bbbe-4b80-909b-782c0example",
              "negated": false
            },
            "action": "INSERT"
          }
        ]
      },
      "responseElements": {
        "changeToken": "32343a11-35e2-4dab-81d8-6d408example"
      },
      "requestID": "11918283-0b2d-11e6-9ccc-f9921example",
      "eventID": "00032abc-5bce-4237-a8ee-5f1a9example",
      "eventType": "AwsApiCall",
      "apiVersion": "2015-08-24",
      "recipientAccountId": "777777777777"
    },
    {
      "eventVersion": "1.03",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAIEP4IT4TPDEXAMPLE",
        "arn": "arn:aws:iam::777777777777:user/nate",
        "accountId": "777777777777",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "nate"
      },
      "eventTime": "2016-04-25T21:35:28Z",
      "eventSource": "waf.amazonaws.com",
      "eventName": "DeleteRule",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "AWS Internal",
      "userAgent": "console.amazonaws.com",
      "requestParameters": {
        "changeToken": "fd232003-62de-4ea3-853d-52932example",
        "ruleId": "3e3e2d11-fd8b-4333-8b03-1da95example"
      },
      "responseElements": {
        "changeToken": "fd232003-62de-4ea3-853d-52932example"
      },
      "requestID": "b23458a1-0b2d-11e6-9ccc-f9928example",
      "eventID": "a3236565-1a1a-4475-978e-81c12example",
      "eventType": "AwsApiCall",
      "apiVersion": "2015-08-24",
      "recipientAccountId": "777777777777"
    }
  ]
}
```

# Responding to DDoS Attacks

Layer 3 and layer 4 attacks are addressed automatically by AWS. However, if DDoS alarms in CloudWatch indicate a possible layer 7 attack, you have two options:

- Investigate and mitigate the attack on your own: If you determine that activity represents a DDoS attack, you can create your own AWS WAF rules to mitigate the attack. AWS WAF is included with AWS Shield Advanced at no additional cost. AWS provides preconfigured templates to get you started quickly. The templates include a set of AWS WAF rules, which are designed to block common web-based attacks. You can customize the rules to fit your business needs. For more information, see AWS WAF Security Automations and Creating a Web ACL (p. 53).
- If you are an AWS Shield Advanced customer, you also have the option of contacting the AWS Support Center: If you want assistance in applying mitigations, you can contact the AWS Support Center. Critical and urgent cases are routed directly to DDoS experts. With AWS Shield Advanced, complex cases can be escalated to the DRT, which has deep experience in protecting AWS, Amazon.com, and its subsidiaries.

  To get DRT support, contact the AWS Support Center and explain that you are an AWS Shield Advanced customer experiencing a possible attack. Our representative will direct your call to the appropriate DDoS experts. If you open a case with the AWS Support Center using the **Distributed Denial of Service (DDoS)** service type, you can speak directly with a DDoS expert by chat or telephone. DDoS support engineers can help you identify attacks, recommend improvements to your AWS architecture, and provide guidance in the use of AWS services for DDoS attack mitigation.

  > **Important**
  > For layer 7 attacks, the DRT can help you analyze the suspicious activity, and then assist you to mitigate the issue. This mitigation often requires the DRT to create or update AWS WAF web access control lists (web ACLs) in your account. However, they need your permission to do so. We recommend that as part of enabling AWS Shield Advanced, you follow the steps in Step 3: Authorize the DDoS Response Team to Create Rules and Web ACLs on Your Behalf (p. 89) to proactively provide the DRT with the needed permissions. Providing permission ahead of time helps to prevent any delays in the event of an actual attack.

## Reviewing DDoS Incidents

AWS Shield Advanced provides real-time metrics and reports for extensive visibility into attacks on your AWS resources. Details about active and past incidents that have occurred in the last 12 months include attack type, start time, and duration. For a description of each of the attack types, see Types of DDoS Attacks (p. 83). Choose a specific incident to review additional details.

These metrics and reports are available only for AWS Shield Advanced customers. To activate AWS Shield Advanced, see To enable and configure AWS Shield Advanced (p. 87).

### To review DDoS incidents

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/waf/.
2. Choose **Incidents**.
3. Choose the **Incident type** of the attack you want to investigate.

If you determine a possible attack is underway, you can contact the DRT through the AWS Support Center, or attempt to mitigate the attack on your own by creating a new web access control list (web ACL).

### To mitigate a potential DDoS attack

1. Create conditions in AWS WAF that match the unusual behavior.
2. Add those conditions to one or more AWS WAF rules.
3. Add those rules to a web ACL and configure the web ACL to count the requests that match the rules.
4. Monitor those counts to determine if the source of the requests should be blocked. If the volume of requests continue to be unusually high, change your web ACL to block those requests.

   For more information, see Creating a Web ACL (p. 53).

AWS provides preconfigured templates to get you started quickly. The templates include a set of AWS WAF rules, which can be customized to best fit your needs, designed to block common web-based attacks. For more information, see AWS WAF Security Automations.

# Using the AWS WAF and AWS Shield Advanced API

This section describes how to make requests to the AWS WAF and Shield Advanced API for creating and managing match sets, rules, and web ACLs in AWS WAF as well as your subscription and protections in Shield Advanced. This section will acquaint you with the components of requests, the content of responses, and how to authenticate requests.

Topics

## Using the AWS SDKs

If you use a language that AWS provides an SDK for, use the SDK rather than trying to work your way through the APIs. The SDKs make authentication simpler, integrate easily with your development environment, and provide easy access to AWS WAF and Shield Advanced commands. For more information about the AWS SDKs, see Step 3: Download Tools (p. 5) in the topic Setting Up for AWS WAF and AWS Shield (p. 3).

## Making HTTPS Requests to AWS WAF or Shield Advanced

AWS WAF and Shield Advanced requests are HTTPS requests, as defined by RFC 2616. Like any HTTP request, a request to AWS WAF or Shield Advanced contains a request method, a URI, request headers, and a request body. The response contains an HTTP status code, response headers, and sometimes a response body.

# Request URI

The request URI is always a single forward slash, `/`.

# HTTP Headers

AWS WAF and Shield Advanced require the following information in the header of an HTTP request:

**Host (Required)**

The endpoint that specifies where your resources are created. The various endpoints can be found in AWS Regions and Endpoints. For example, the value of the `Host` header for AWS WAF for a CloudFront distribution is `waf.amazonaws.com:443`.

**x-amz-date or Date (Required)**

The date used to create the signature that is contained in the `Authorization` header. Specify the date in ISO 8601 standard format, in UTC time, as shown in the following example:

```
x-amz-date: 20151007T174952Z
```

You must include either `x-amz-date` or `Date`. (Some HTTP client libraries don't let you set the `Date` header). When an `x-amz-date` header is present, AWS WAF ignores any `Date` header when authenticating the request.

The time stamp must be within 15 minutes of the AWS system time when the request is received. If it isn't, the request fails with the `RequestExpired` error code to prevent someone else from replaying your requests.

**Authorization (Required)**

The information required for request authentication. For more information about constructing this header, see Authenticating Requests (p. 104).

**X-Amz-Target (Required)**

A concatenation of `AWSWAF_` or `AWSShield_`, the API version without punctuation, a period (`.`), and the name of the operation, for example:

```
AWSWAF_20150824.CreateWebACL
```

**Content-Type (Conditional)**

Specifies that the content type is JSON as well as the version of JSON, as shown in the following example:

```
Content-Type: application/x-amz-json-1.1
```

Condition: Required for POST requests.

**Content-Length (Conditional)**

Length of the message (without the headers) according to RFC 2616.

Condition: Required if the request body itself contains information (most toolkits add this header automatically).

The following is an example header for an HTTP request to create a web ACL in AWS WAF:

```
POST / HTTP/1.1
Host: waf.amazonaws.com:443
```

```
X-Amz-Date: 20151007T174952Z
Authorization: AWS4-HMAC-SHA256
               Credential=AccessKeyID/20151007/us-east-2/waf/aws4_request,
               SignedHeaders=host;x-amz-date;x-amz-target,
               Signature=145b1567ab3c50d929412f28f52c45dbf1e63ec5c66023d232a539a4afd11fd9
X-Amz-Target: AWSWAF_20150824.CreateWebACL
Accept: */*
Content-Type: application/x-amz-json-1.1; charset=UTF-8
Content-Length: 231
Connection: Keep-Alive
```

## HTTP Request Body

Many AWS WAF and Shield Advanced API actions require you to include JSON-formatted data in the body of the request.

The following example request uses a simple JSON statement to update an `IPSet` (known in the console as an IP match condition) to include the IP address 192.0.2.44 (represented in CIDR notation as 192.0.2.44/32):

```
POST / HTTP/1.1
Host: waf.amazonaws.com:443
X-Amz-Date: 20151007T174952Z
Authorization: AWS4-HMAC-SHA256
               Credential=AccessKeyID/20151007/us-east-2/waf/aws4_request,
               SignedHeaders=host;x-amz-date;x-amz-target,
               Signature=145b1567ab3c50d929412f28f52c45dbf1e63ec5c66023d232a539a4afd11fd9
X-Amz-Target: AWSWAF_20150824.UpdateIPSet
Accept: */*
Content-Type: application/x-amz-json-1.1; charset=UTF-8
Content-Length: 283
Connection: Keep-Alive

{
   "ChangeToken": "d4c4f53b-9c7e-47ce-9140-0ee5ffffffff",
   "IPSetId": "69d4d072-170c-463d-ab82-0643ffffffff",
   "Updates": [
      {
         "Action": "INSERT",
         "IPSetDescriptor": {
            "Type": "IPV4",
            "Value": "192.0.2.44/32"
         }
      }
   ]
}
```

# HTTP Responses

All AWS WAF and Shield Advanced API actions include JSON-formatted data in the response.

Here are some important headers in the HTTP response and how you should handle them in your application, if applicable:

**HTTP/1.1**

This header is followed by a status code. Status code 200 indicates a successful operation.

Type: String

**x-amzn-RequestId**

A value created by AWS WAF or Shield Advanced that uniquely identifies your request, for example, `K2QH8DNOU907N97FNA2GDLL8OBVV4KQNSO5AEMVJF66Q9ASUAAJG`. If you have a problem with AWS WAF, AWS can use this value to troubleshoot the problem.

Type: String

**Content-Length**

The length of the response body in bytes.

Type: String

**Date**

The date and time that AWS WAF or Shield Advanced responded, for example, Wed, 07 Oct 2015 12:00:00 GMT.

Type: String

## Error Responses

If a request results in an error, the HTTP response contains the following values:

- A JSON error document as the response body
- `Content-Type header: text/xml`
- The applicable 3xx, 4xx, or 5xx HTTP status code

The following is an example of a JSON error document:

```
HTTP/1.1 400 Bad Request
x-amzn-RequestId: b0e91dc8-3807-11e2-83c6-5912bf8ad066
x-amzn-ErrorType: ValidationException
Content-Type: application/json
Content-Length: 125
Date: Mon, 26 Nov 2012 20:27:25 GMT

{"message":"1 validation error detected: Value null at 'TargetString' failed to satisfy
 constraint: Member must not be null"}
```

# Authenticating Requests

If you use a language that AWS provides an SDK for, we recommend that you use the SDK. All the AWS SDKs greatly simplify the process of signing requests and save you a significant amount of time when compared with using the AWS WAF or Shield Advanced API. In addition, the SDKs integrate easily with your development environment and provide easy access to related commands.

AWS WAF and Shield Advanced require that you authenticate every request that you send by signing the request. To sign a request, you calculate a digital signature using a cryptographic hash function, which returns a hash value based on the input. The input includes the text of your request and your secret access key. The hash function returns a hash value that you include in the request as your signature. The signature is part of the `Authorization` header of your request.

After receiving your request, AWS WAF or Shield Advanced recalculates the signature using the same hash function and input that you used to sign the request. If the resulting signature matches the signature in the request, AWS WAF or Shield Advanced processes the request. If not, the request is rejected.

AWS WAF and Shield Advanced supports authentication using AWS Signature Version 4. The process for calculating a signature can be broken into three tasks:

### Task 1: Create a Canonical Request

Create your HTTP request in canonical format as described in Task 1: Create a Canonical Request For Signature Version 4 in the *Amazon Web Services General Reference*.

### Task 2: Create a String to Sign

Create a string that you will use as one of the input values to your cryptographic hash function. The string, called the string to sign, is a concatenation of the following values:

- Name of the hash algorithm
- Request date
- Credential scope string
- Canonicalized request from the previous task

The credential scope string itself is a concatenation of date, region, and service information.

For the `X-Amz-Credential` parameter, specify the following:

- The code for the endpoint to which you're sending the request, `us-east-2`
- `waf` for the service abbreviation

For example:

```
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20130501/us-east-2/waf/aws4_request
```

### Task 3: Create a Signature

Create a signature for your request by using a cryptographic hash function that accepts two input strings:

- Your string to sign, from Task 2.
- A derived key. The derived key is calculated by starting with your secret access key and using the credential scope string to create a series of hash-based message authentication codes (HMACs).

# AWS WAF PCI DSS Compliance

AWS WAF is now a Payment Card Industry (PCI) Data Security Standard (DSS) 3.2 compliant service. The PCI Standards Council published PCI DSS version 3.2 in April 2016 as the most updated set of requirements available. PCI DSS 3.2 has revised and clarified the online credit card transaction requirements for encryption, access control, change management, application security, and risk management programs.

For more information about AWS and PCI DSS 3.2 compliance, see AWS Becomes First Cloud Service Provider to Adopt New PCI DSS 3.2 on the AWS Security Blog. For more information about PCI DSS version 3.2, see PCI DSS 3.2: What's New?.

# Resources

The following related resources can help you as you work with this service.

## AWS Resources

Several helpful guides, forums, and other resources are available from Amazon Web Services.

- **AWS WAF Discussion Forum** – A community-based forum for developers to discuss technical questions related to AWS WAF.
- **Shield Advanced Discussion Forum** – A community-based forum for developers to discuss technical questions related to Shield Advanced.
- **AWS WAF product information** – The primary web page for information about AWS WAF, including features, pricing, and more.
- **Shield Advanced product information** – The primary web page for information about Shield Advanced, including features, pricing, and more.

- **Classes & Workshops** – Links to role-based and specialty courses as well as self-paced labs to help sharpen your AWS skills and gain practical experience.
- **AWS Developer Tools** – Links to developer tools, SDKs, IDE toolkits, and command line tools for developing and managing AWS applications.
- **AWS Whitepapers** – Links to a comprehensive list of technical AWS whitepapers, covering topics such as architecture, security, and economics and authored by AWS Solutions Architects or other technical experts.
- **AWS Support Center** – The hub for creating and managing your AWS Support cases. Also includes links to other helpful resources, such as forums, technical FAQs, service health status, and AWS Trusted Advisor.
- **AWS Support** – The primary web page for information about AWS Support, a one-on-one, fast-response support channel to help you build and run applications in the cloud.
- **Contact Us** – A central contact point for inquiries concerning AWS billing, account, events, abuse, and other issues.
- **AWS Site Terms** – Detailed information about our copyright and trademark; your account, license, and site access; and other topics.

# Document History

- **API Version:** 2015-08-24
- **Latest documentation update:** July 25, 2016

The following table describes important changes in each release of the *AWS WAF Developer Guide*.

| Change | API Version | Description | Release Date |
|---|---|---|---|
| Update | 2016-08-24 | Rate-based rules | June, 2017 |
| Update | 2016-08-24 | Reorganization | April, 2017 |
| Update | 2016-08-24 | Added information about DDOS protection and support for Application Load Balancers. | November, 2016 |
| Update | 2015-08-24 | Added information about AWS WAF PCI DSS Compliance (p. 106) for AWS WAF. | July 25, 2016 |
| New Features | 2015-08-24 | You can now log all your API calls to AWS WAF through AWS CloudTrail, the AWS service that records API calls for your account and delivers log files to your S3 bucket. CloudTrail logs can be used to enable security analysis, track changes to your AWS resources, and aid in compliance auditing. Integrating AWS WAF and CloudTrail lets you determine which requests were made to the AWS WAF API, the source IP address from which each request was made, who made the request, when it was made, and more.<br>If you are already using AWS CloudTrail, you will start seeing AWS WAF API calls in your AWS CloudTrail log. If you haven't turned on AWS CloudTrail for your account, you can turn on CloudTrail from the AWS Management Console. There is no additional charge for turning on CloudTrail, but standard rates for Amazon S3 and Amazon SNS usage apply. | April 28, 2016 |

| Change | API Version | Description | Release Date |
|---|---|---|---|
| New Features | 2015-08-24 | You can now use AWS WAF to allow, block, or count web requests that appear to contain malicious scripts, known as cross-site scripting or XSS. Attackers sometimes insert malicious scripts into web requests in an effort to exploit vulnerabilities in web applications. For more information, see Working with Cross-site Scripting Match Conditions (p. 31). | March 29, 2016 |
| New Features | 2015-08-24 | With this release, AWS WAF adds the following features:<br>• You can configure AWS WAF to allow, block, or count web requests based on the lengths of specified parts of the requests, such as query strings or URIs. For more information, see Working with Size Constraint Conditions (p. 37).<br>• You can configure AWS WAF to allow, block, or count web requests based on the content in the request body. This is the part of a request that contains any additional data that you want to send to your web server as the HTTP request body, such as data from a form. This feature applies to string match conditions, SQL injection match conditions, and the new size constraint conditions mentioned in the first bullet. For more information, see the following documentation:<br>  • Values That You Specify When You Create or Edit String Match Conditions (p. 46)<br>  • Values That You Specify When You Create or Edit SQL Injection Match Conditions (p. 43)<br>  • Values That You Specify When You Create or Edit Size Constraint Conditions (p. 39) | January 27, 2016 |
| New Feature | 2015-08-24 | You can now use the AWS WAF console to choose the CloudFront distributions that you want to associate a web ACL with. For more information, see Associating or Disassociating a Web ACL and a CloudFront Distribution. | November 16, 2015 |
| Initial Release | 2015-08-24 | This is the first release of the *AWS WAF Developer Guide*. | October 6, 2015 |

# AWS Glossary

For the latest AWS terminology, see the AWS Glossary in the *AWS General Reference*.