AWS WAF, AWS Firewall Manager, and AWS Shield Advanced

Developer Guide

AWS WAF, AWS Firewall Manager, and AWS Shield Advanced: Developer Guide

Table of Contents

What are AWS WAF, AWS Shield, and AWS Firewall Manager?

AWS WAF is a web application firewall that lets you monitor the HTTP and HTTPS requests that are forwarded to an Amazon CloudFront distribution, an Amazon API Gateway REST API, an Application Load Balancer, or an AWS AppSync GraphQL API. 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, Amazon CloudFront, Amazon API Gateway, Application Load Balancer, or AWS AppSync 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 Amazon CloudFront, Amazon API Gateway, Application Load Balancer, or AWS AppSync 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.
  - Country that requests originate from.
  - Values in request headers.
  - Strings that appear in requests, either specific strings or strings that match regular expression (regex) patterns.
  - 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.
- Managed rule groups from AWS and AWS Marketplace sellers.
- 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. For additional protection against DDoS attacks, 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. AWS Shield Advanced provides expanded DDoS attack protection for your Amazon EC2 instances, Elastic Load Balancing load balancers, CloudFront distributions, Route 53 hosted zones, and AWS Global Accelerator accelerators. AWS Shield Advanced incurs additional charges.

For more information about AWS Shield Standard and AWS Shield Advanced, see AWS Shield (p. 325).

AWS Firewall Manager

AWS Firewall Manager simplifies your administration and maintenance tasks across multiple accounts and resources for AWS WAF rules, AWS Shield Advanced protections, and Amazon VPC security groups. The Firewall Manager service automatically applies your rules and other security protections across your accounts and resources, even as you add new accounts and resources.

For more information about Firewall Manager, see AWS Firewall Manager (p. 249).

Which should I choose?

You can use AWS WAF (p. 6), AWS Firewall Manager (p. 249), and AWS Shield (p. 325) together to create a comprehensive security solution.

It all starts with AWS WAF. You can automate and then simplify AWS WAF management using AWS Firewall Manager. Shield Advanced adds additional features on top of AWS WAF, such as dedicated support from the Shield Response Team (SRT) and advanced reporting.

If you want granular control over the protection that is added to your resources, AWS WAF alone is the right choice. If you want to use AWS WAF across accounts, accelerate your AWS WAF configuration, or automate protection of new resources, use Firewall Manager with AWS WAF.

Finally, if you own high visibility websites or are otherwise prone to frequent DDoS attacks, you should consider purchasing the additional features that Shield Advanced provides.

> **Note**
> To use the services of the SRT, you must be subscribed to the Business Support plan or the Enterprise Support plan.

Setting up

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

After you complete these steps, see Getting started with AWS WAF (p. 8) to continue getting started with AWS WAF.

> **Note**
> AWS Shield Standard is included with AWS WAF and does not require additional setup. For more information, see How AWS Shield works (p. 325).

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. 3)
- 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://portal.aws.amazon.com/billing/signup.
2. Follow the online instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a verification code on 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 administrative permissions directly to the IAM user account.
- Verify that the account has full access to AWS WAF and related services, for general use and for console access. For information, see AWS managed (predefined) policies for AWS WAF (p. 124).

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 administrator user for yourself and add the user to an administrators group (console)

1. Sign in to the IAM console as the account owner by choosing **Root user** and entering your AWS account email address. On the next page, enter your password.

   > **Note**
   > We strongly recommend that you adhere to the best practice of using the **`Administrator`** IAM user that follows and securely lock away the root user credentials. Sign in as the root user only to perform a few account and service management tasks.
2. In the navigation pane, choose **Users** and then choose **Add user**.
3. For **User name**, enter **`Administrator`**.
4. Select the check box next to **AWS Management Console access**. Then select **Custom password**, and then enter your new password in the text box.
5. (Optional) By default, AWS requires the new user to create a new password when first signing in. You can clear the check box next to **User must create a new password at next sign-in** to allow the new user to reset their password after they sign in.
6. Choose **Next: Permissions**.
7. Under **Set permissions**, choose **Add user to group**.
8. Choose **Create group**.
9. In the **Create group** dialog box, for **Group name** enter **`Administrators`**.
10. Choose **Filter policies**, and then select **AWS managed - job function** to filter the table contents.
11. In the policy list, select the check box for **AdministratorAccess**. Then choose **Create group**.

    > **Note**
    > You must activate IAM user and role access to Billing before you can use the `AdministratorAccess` permissions to access the AWS Billing and Cost Management console. To do this, follow the instructions in step 1 of the tutorial about delegating access to the billing console.
12. 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.
13. Choose **Next: Tags**.
14. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information about using tags in IAM, see Tagging IAM entities in the *IAM User Guide*.
15. 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 that restrict user permissions to specific AWS resources, see Access management and Example policies.

To sign in as this new IAM user, first sign out of the AWS Management 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.

After you complete these steps, you can stop here and go to Getting started with AWS WAF (p. 8) to continue getting started with AWS WAF using the console. If you want to access AWS WAF programmatically using the AWS WAF API, continue on to the next step, Step 3: Download tools (p. 5).

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 PowerShell Cmdlet Reference.

AWS WAF

AWS WAF is a web application firewall that lets you monitor the HTTP(S) requests that are forwarded to an Amazon CloudFront distribution, an Amazon API Gateway REST API, an Application Load Balancer, or an AWS AppSync GraphQL API.

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, the service associated with your protected resource responds to requests either with the requested content or with an HTTP 403 status code (Forbidden). You can also configure CloudFront to return a custom error page when a request is blocked.

> **Note**
> You can also use AWS WAF to protect your applications that are hosted in Amazon Elastic Container Service (Amazon ECS) containers. Amazon ECS is a highly scalable, fast container management service that makes it easy to run, stop, and manage Docker containers on a cluster. To use this option, you configure Amazon ECS to use an Application Load Balancer that is enabled for AWS WAF to route and protect HTTP(S) layer 7 traffic across the tasks in your service. For more information, see Service Load Balancing in the *Amazon Elastic Container Service Developer Guide*.

Topics

How AWS WAF works

You use AWS WAF to control how an Amazon CloudFront distribution, an Amazon API Gateway REST API, an Application Load Balancer, or an AWS AppSync GraphQL API responds to HTTP(S) web requests.

- **Web ACLs** – You use a web access control list (ACL) to protect a set of AWS resources. You create a web ACL and define its protection strategy by adding rules. Rules define criteria for inspecting web requests and specify how to handle requests that match the criteria. You set a default action for the web ACL that indicates whether to block or allow through those requests that pass the rules inspections.

- **Rules** – Each rule contains a statement that defines the inspection criteria, and an action to take if a web request meets the criteria. When a web request meets the criteria, that's a match. You can use rules to block matching requests or to allow matching requests through. You can also use rules just to count matching requests.
- **Rules groups** – You can use rules individually or in reusable rule groups. AWS Managed Rules and AWS Marketplace sellers provide managed rule groups for your use. You can also define your own rule groups.

After you create your web ACL, you can associate it with one or more AWS resources. The resource types that you can protect using AWS WAF web ACLs are an Amazon CloudFront distribution, an Amazon API Gateway REST API, an Application Load Balancer, and an AWS AppSync GraphQL API.

AWS WAF is available in the Regions listed at AWS service endpoints.

- For an Amazon API Gateway REST API, an Application Load Balancer, or an AWS AppSync GraphQL API, you can use any of the Regions in the list.
- For a CloudFront distribution, AWS WAF is available globally, but you must use the Region US East (N. Virginia) for all of your work. You must create your web ACL using the Region US East (N. Virginia). You must also use this Region to create any other resources that you use in your web ACL, like rule groups, IP sets, and regex pattern sets.

  Some interfaces offer a region choice of "Global (CloudFront)". Choosing this is identical to choosing Region US East (N. Virginia) or "us-east-1".

AWS WAF Web ACL capacity units (WCU)

AWS WAF uses web ACL capacity units (WCU) to calculate and control the operating resources that are required to run your rules, rule groups, and web ACLs. AWS WAF enforces WCU limits when you configure your rule groups and web ACLs. WCUs don't affect how AWS WAF inspects web traffic.

AWS WAF calculates capacity differently for each rule type, to reflect each rule's relative cost. Simple rules that cost little to run use fewer WCUs than more complex rules that use more processing power. For example, a size constraint rule statement uses fewer WCUs than a statement that inspects against a regex pattern set.

AWS WAF manages capacity for rules, rule groups, and web ACLs:

- **Rule capacity** – AWS WAF calculates rule capacity when you create or update a rule. For some basic guidelines for rule capacity requirements, see the listings for the various rule statements at AWS WAF rule statements (p. 60). You can also get an idea of the capacity required for the various rule types in the AWS WAF console by creating a web ACL or rule group and adding individual rules to it. The console displays the capacity units used as you add the rules.
- **Rule group capacity** – AWS WAF requires that each rule group is assigned an immutable capacity at creation. This is true for managed rule groups and rule groups that you create through AWS WAF. When you modify a rule group, your changes must keep the rule group's WCU within its capacity. This ensures that web ACLs that are using the rule group remain within their maximum capacity.
- **Web ACL capacity** – The maximum capacity for a web ACL is 1,500, which is sufficient for most use cases. If you need more capacity, contact the AWS Support Center.

AWS WAF pricing

With AWS WAF, you pay only for the web ACLs and rule groups that you create, and for the number of HTTP(S) 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 wizard in the AWS WAF console.
- Choose the AWS resources that you want AWS WAF to inspect web requests for. This tutorial covers the steps for Amazon CloudFront. The process is essentially the same for an Amazon API Gateway REST API, an Application Load Balancer, or an AWS AppSync GraphQL API.
- Add the rules and rule groups 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. For each rule, you specify whether you want to block matching web requests or allow them. The rules that are defined inside a rule group have their actions defined inside the rule group.
- Specify a default action for the web ACL, either `Block` or `Allow`. This is the action that AWS WAF takes when a web request doesn't match any of the rules.

> **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 (p. 3), go to Step 2: Create a Web ACL (p. 8).

If not, go to Setting up (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. For more information about AWS WAF web ACLs, see Managing and using a web access control list (web ACL) (p. 17).

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/wafv2/.
2. From the AWS WAF home page, choose **Create web ACL**.

3. For **Name**, enter the name that you want to use to identify this web ACL.

   Note
   You can't change the name after you create the web ACL.
4. (Optional) For **Description - optional**, enter a longer description for the web ACL if you want to.
5. For **CloudWatch metric name**, change the default name if applicable. Follow the guidance on the console for valid characters. The name can't contain special characters, white space, or metric names reserved for AWS WAF, including "All" and "Default_Action."

   Note
   You can't change the CloudWatch metric name after you create the web ACL.
6. For **Resource type**, choose **CloudFront distributions**. The **Region** automatically populates to **Global (CloudFront)** for CloudFront distributions.
7. (Optional) For **Associated AWS resources - optional**, choose **Add AWS resources**. In the dialog box, choose the resources that you want to associate, and then choose **Add**. AWS WAF returns you to the **Describe web ACL and associated AWS resources** page.
8. Choose **Next**.

Step 3: Add a string match rule

In this step, you create a rule with a string match statement and indicate what to do with matching requests. A string match rule statement identifies strings that you want AWS WAF to search for in a request. Usually, a string consists of printable ASCII characters, but you can specify any character from hexadecimal 0x00 to 0xFF (decimal 0 to 255). In addition to specifying the string to search for, you specify the web request component that you want to search, such as a header, a query string, or the request body.

This statement type operates on a web request component, and requires the following request component settings:

- **Request components** – The part of the web request to inspect, for example, a query string or the body.

  Note
  If you use the request component **Body** or **JSON body**, only the first 8 KB (8,192 bytes) of the request body are forwarded to AWS WAF for inspection. AWS WAF does not support inspecting the entire contents of web requests whose bodies exceed 8 KB.

  For information about web request components, see Request component (p. 75).
- **Optional text transformations** – Transformations that you want AWS WAF to perform on the request component before inspecting it. For example, you could transform to lowercase or normalize white space. If you specify more than one transformation, AWS WAF processes them in the order listed. For information, see Text transformations (p. 78).

For additional information about AWS WAF rules, see AWS WAF rules (p. 52).

To create a string match rule statement

1. On the **Add rules and rule groups** page, choose **Add rules**, **Add my own rules and rule groups**, **Rule builder**, then **Rule visual editor**.

   Note
   The console provides the **Rule visual editor** and also a **Rule JSON editor**. The JSON editor makes it easy for you to copy configurations between web ACLs and is required for more complex rule sets, like those with multiple levels of nesting.
   This procedure uses the **Rule visual editor**.

2. For **Name**, enter the name that you want to use to identify this rule.
3. For **Type** choose **Regular rule**.
4. For **If a request** choose **matches the statement**.

   The other options use the logical statement types for rules, which allow you to combine or negate rule statement results.
5. On **Statement**, for **Inspect**, open the dropdown and choose the web request component that you want AWS WAF to look for your string in. For this example, choose **Header**.

   When you choose **Header**, you also specify which header you want AWS WAF to inspect. Enter **`User-Agent`**. This value isn't case sensitive.
6. For **Match type**, choose where the specified string must appear in the `User-Agent` header.

   For this example, choose **Exactly matches string**. This indicates that AWS WAF inspects the user-agent header in each web request for a string that is identical to the string that you specify.
7. For **String to match**, specify a string that you want AWS WAF to search for. The maximum length of **String to match** is 200 characters. If you want to specify a base64-encoded value, you can specify up to 200 characters before encoding.

   For this example, enter **MyAgent**. AWS WAF will inspect the `User-Agent` header in web requests for the value `MyAgent`.
8. Leave **Text transformation** set to **None**.
9. For **Action**, select the action that you want the rule to take when it matches a web request. For this example, choose **Count** and leave the other choices as they are. The count action creates metrics for web requests that match the rule, but doesn't affect whether the request is allowed or blocked. For more information about action choices, see AWS WAF rule action (p. 53) and Web ACL rule and rule group evaluation (p. 19).
10. Choose **Add rule**.

Step 4: Add an AWS Managed Rules rule group

AWS Managed Rules offers a set of managed rule groups for your use, most of which are free of charge to AWS WAF customers. For more information about rule groups, see Rule groups (p. 30). We'll add an AWS Managed Rules rule group to this web ACL.

To add an AWS Managed Rules rule group

1. On the **Add rules and rule groups** page, choose **Add rules**, and then choose **Add managed rule groups**.
2. On the **Add managed rule groups** page, expand the listing for the **AWS managed rule groups**. (You'll also see listings offered for AWS Marketplace sellers. You can subscribe to their offerings and then use them in the same way as for AWS Managed Rules rule groups.)
3. For the rule group that you want to add, do the following:

   a. In the **Action** column, turn on the **Add to web ACL** toggle.

   b. Select **Edit** and, in the rule group's **Rules** listing, turn on the **Set all rule actions to count** toggle. This sets the action for all rules in the rule group to count only. This allows you to see how all of the rules in the rule group behave with your web requests before you put any of them to use.

   c. Choose **Save rule**.
4. In the **Add managed rule groups** page, choose **Add rules**. This returns you to the **Add rules and rule groups** page.

Step 5: Finish your Web ACL configuration

When you're done adding rules and rule groups to your web ACL configuration, finish up by managing the priority of the rules in the web ACL and configuring settings like metrics, tagging, and logging.

To finish your web ACL configuration

1. On the **Add rules and rule groups** page, choose **Next**.
2. On the **Set rule priority** page, you can see the processing order for the rules and rule groups in the web ACL. AWS WAF processes them starting from the top. You can change the processing order by moving them up and down. To do this, select one in the list and choose **Move up** or **Move down**.
3. Choose **Next**.
4. On the **Configure metrics** page, for **Amazon CloudWatch metrics**, you can see the planned metrics for your rules and rule groups and you can see the web request sampling options. For information about Amazon CloudWatch metrics, see Monitoring with Amazon CloudWatch (p. 370). For information about viewing sampled requests, see Viewing a sample of web requests (p. 28).
5. Choose **Next**.
6. On the **Review and create web ACL** page, review your settings, then choose **Create web ACL**.

The wizard returns you to the **Web ACL** page, where your new web ACL is listed.

Step 6: Clean up your resources

You've now successfully completed the tutorial. To prevent your account from accruing additional AWS WAF charges, 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. In the **Web ACL** page, select your web ACL from the list and choose **Edit**.
2. On **Associated AWS resources - optional**, select all associated resources, and then choose **Remove**. This disassociates the web ACL from your AWS resources.
3. In each of the following screens, choose **Next** until you return to the **Web ACL** page.

   In the **Web ACL** page, select your web ACL from the list and choose **Delete**.

Rules and rule statements don't exist outside of rule group and web ACL definitions. If you delete a web ACL, this deletes all individual rules that you've defined in the web ACL. When you remove a rule group from a web ACL, you just remove the reference to it.

Migrating your AWS WAF Classic resources to AWS WAF

This section provides guidance for migrating your rules and web ACLs from AWS WAF Classic to AWS WAF. AWS WAF was released in November 2019. If you created resources like rules and web ACLs using AWS WAF Classic, you either need to work with them using AWS WAF Classic or migrate them to this latest version.

Before you start your migration work, familiarize yourself with AWS WAF by reading through AWS WAF (p. 6).

Topics

Why migrate to AWS WAF?

The latest version of AWS WAF provides many improvements over the prior version, while maintaining most of the concepts and terminology that you're accustomed to.

The following list describes the major changes in the latest AWS WAF. Before you continue with your migration, please take some time to review this list and to familiarize yourself with the rest of the AWS WAF guide.

- **AWS Managed Rules for AWS WAF** – The rule groups now available through AWS Managed Rules provide protection against common web threats. Most of these rule groups are included free of charge with AWS WAF. For more information, see AWS Managed Rules rule groups list (p. 38) and the blog post Announcing AWS Managed Rules for AWS WAF.
- **New AWS WAF API** – The new API allows you to configure all of your AWS WAF resources using a single set of APIs. To distinguish between regional and global applications, the new API includes a `scope` setting. For more information about the API, see the AWS WAFV2 Actions and AWS WAFV2 Data Types.

  In the APIs, SDKs, CLIs, and AWS CloudFormation, AWS WAF Classic retains its naming schemes and this latest version of AWS WAF is referred to with an added `V2` or `v2`, depending on the context.
- **Simplified service quotas (limits)** – AWS WAF now allows more rules per web ACL and allows you to express longer regex patterns. For more information, see AWS WAF quotas (p. 135).
- **Web ACL limits are now based on computing needs** – Web ACL limits are now based on Web ACL capacity units (WCU). AWS WAF calculates the WCU for a rule according to the operating capacity that's required to run the rule. The WCU of a web ACL is the sum of the WCU of all rules and rule groups in the web ACL.

  For general information about WCU, see How AWS WAF works (p. 6). For information about each rule's WCU usage, see Rule statements list (p. 61).
- **Document-based rule writing** – You can now write and express rules, rule groups, and web ACLs in JSON format. You no longer need to use individual API calls to create different conditions and then associate the conditions to a rule. This greatly simplifies how you write and maintain your code. You can access a JSON format of your web ACLs through the console when you're viewing the web ACL, by choosing **Download web ACL as JSON**. When you are creating your own rule, you can access its JSON representation by choosing **Rule JSON editor**.
- **Rule nesting and full logical operation support** – You can write complex combined rules by using logical rule statements and by using nesting. You can create statements such as `[A AND NOT(B OR C)]`. For more information, see Rule statements list (p. 61).
- **Variable CIDR range support for IP set** – IP set specifications now have more flexibility in the IP ranges. For IPv4, AWS WAF supports `/1` to `/32`. For IPv6, AWS WAF supports `/1` to `/128`. For more information about IP sets, see IP set match rule statement (p. 65).
- **Chainable text transformations** – AWS WAF can perform multiple text transformations against web request content before inspecting it. For more information, see Text transformations (p. 78).
- **Improved console experience** – The new AWS WAF console features visual rule builder and a more user intuitive console design.

- **Expanded options for Firewall Manager AWS WAF policies** – In the Firewall Manager management of AWS WAF web ACLs, you can now create a set of rule groups that AWS WAF processes first and a set of rule groups that AWS WAF processes last. After you apply the AWS WAF policy, local account owners can add their own rule groups that AWS WAF processes in between these two sets. For more information about Firewall Manager AWS WAF policies, see AWS WAF policies (p. 284).
- **AWS CloudFormation support for all rule statement types** – AWS WAF in AWS CloudFormation supports all rule statement types that the AWS WAF console and API support. Additionally, you can easily convert the rules that you write in JSON format to YAML format.

How the migration works

The automated migration carries over most of your AWS WAF Classic web ACL configuration, leaving a few things that you need to handle manually.

The following lists the high-level steps for migrating a web ACL.

1. The automated migration reads everything related to your existing web ACL, without modifying or deleting anything in AWS WAF Classic. It creates a representation of the web ACL and its related resources, compatible with AWS WAF. It generates an AWS CloudFormation template for the new web ACL and stores it in an Amazon S3 bucket.
2. You deploy the template into AWS CloudFormation, in order to recreate the web ACL and related resources in AWS WAF.
3. You review the web ACL, and manually complete the migration, making sure that your new web ACL takes full advantage of the capabilities of the latest AWS WAF.
4. You manually switch your protected resources over to the new web ACL.

Migration caveats and limitations

The migration doesn't carry over all of your settings, exactly as you have them in AWS WAF Classic. A few things, like managed rules, don't map exactly between the two versions. Other settings, like the web ACL's associations with protected AWS resources, are disabled initially in the new version so you can add them when you're ready.

The following list describes the caveats of the migration and describes any steps you might want to take in response. Use this overview to plan your migration. The detailed migration steps, later on, walk you through the recommended mitigation steps.

- **Single account** – You can only migrate AWS WAF Classic resources for any account to AWS WAF resources for the same account.
- **Rate-based rules** – For rate-based rules, the migration doesn't bring over any associated conditions. If you have a rate-based rule with added conditions, recreate the conditions in the migrated web ACL. In AWS WAF, you do this by adding a nested statement in the rate-based rule to narrow the scope of the rule. For more information about rate-based rules in AWS WAF, see Rate-based rule statement (p. 68).
- **Managed rules** – The migration doesn't bring over any managed rules from AWS Marketplace sellers. Some AWS Marketplace sellers have equivalent managed rules for AWS WAF that you can subscribe to again. Before you do this, review the AWS Managed Rules that are provided with the latest version of AWS WAF. Most of these are free of charge for AWS WAF users. For information about managed rules, see Managed rule groups (p. 30).
- **Web ACL associations** – The migration doesn't bring over any associations between the web ACL and protected resources. This is by design, to avoid affecting your production workload. After you verify that everything is migrated correctly, associate the new web ACL with your resources.

- **Logging** – Logging for the migrated web ACL is disabled by default. This is by design. Enable logging when you are ready to switch over from AWS WAF Classic to AWS WAF.
- **AWS Firewall Manager rule groups** – The migration doesn't handle rule groups that are managed by Firewall Manager. You can migrate a web ACL that's managed by Firewall Manager, but the migration doesn't bring over the rule group. Instead of using the migration tool for these web ACLs, recreate the policy for the new AWS WAF in Firewall Manager.

  Note
  The rule groups that Firewall Manager managed for AWS WAF Classic were Firewall Manager rule groups. With the new version of AWS WAF, the rule groups are AWS WAF rule groups. Functionally, they are the same.
- **AWS WAF Security Automations** – Don't try to migrate any AWS WAF Security Automations. The migration doesn't convert Lambda functions, which might be in use by the automations. When a new AWS WAF Security Automations solution is available that's compatible with the latest AWS WAF, redeploy that solution.

Migrating a web ACL from AWS WAF Classic to AWS WAF

To migrate a web ACL and switch over to it, perform the automated migration, then complete a series of manual steps.

Topics

Migrating a web ACL: automated migration

To automatically migrate a web ACL configuration from AWS WAF Classic to AWS WAF

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. Choose **Switch to AWS WAF Classic** and review your configuration settings for the web ACL. Make note of the settings, considering the caveats and limitations described in the preceding section, Migration caveats and limitations (p. 13).
3. In the informational dialogue at the top, locate the sentence that starts with **Migrate web ACLs** and choose the link to the **migration wizard**. This launches the migration wizard.

   If you don't see the informational dialogue, you might have closed it since you launched the AWS WAF Classic console. In the navigation bar, choose **Switch to new AWS WAF** then choose **Switch to AWS WAF Classic**, and the informational dialogue should reappear.
4. Select the web ACL that you want to migrate.
5. For **Migration configuration**, provide an Amazon S3 bucket to use for the template. You need an Amazon S3 bucket that's configured properly for the migration API, to store the AWS CloudFormation template that it generates.

   - If the bucket is encrypted, the encryption must use Amazon S3 (SSE-S3) keys. The migration doesn't support encryption with AWS Key Management Service (SSE-KMS) keys.
   - The bucket name must start with `aws-waf-migration-`. For example, `aws-waf-migration-my-web-acl`.

- The bucket must be in the Region where you are deploying the template. For example, for a web ACL in `us-west-2`, you must use an Amazon S3 bucket in `us-west-2` and you must deploy the template stack to `us-west-2`.

6. For **S3 bucket policy**, we recommend choosing **Auto apply the bucket policy required for migration**. Alternatively, if you want to manage the bucket on your own, you must manually apply the following bucket policy:

   - For global Amazon CloudFront applications (`waf`):

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "apiv2migration.waf.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::<BUCKET_NAME>/AWSWAF/<CUSTOMER_ACCOUNT_ID>/*"
        }
    ]
}
```

   - For regional Amazon API Gateway or Application Load Balancer applications (`waf-regional`):

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "apiv2migration.waf-regional.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::<BUCKET_NAME>/AWSWAF/<CUSTOMER_ACCOUNT_ID>/*"
        }
    ]
}
```

7. For **Choose how to handle rules that cannot be migrated**, choose either to exclude rules that can't be migrated, or to stop the migration. For information about rules that can't be migrated, see Migration caveats and limitations (p. 13).
8. Choose **Next**.
9. For **Create AWS CloudFormation template**, verify your settings, then choose **Start creating AWS CloudFormation template** to begin the migration process. This can take a few minutes, depending on the complexity of your web ACL.
10. In **Create and run AWS CloudFormation stack to complete migration**, you can choose to go to the AWS CloudFormation console to create a stack from the template, to create the new web ACL and its resources. To do this, choose **Create AWS CloudFormation stack**.

After the automatic migration process completes, you're ready to proceed to the manual follow-up steps. See Migrating a web ACL: manual follow-up (p. 15).

Migrating a web ACL: manual follow-up

After the automated migration is complete, review the newly created web ACL and fill in the components that the migration doesn't bring over for you. The following procedure covers the aspects

of web ACL management that the migration doesn't handle. For the list, see Migration caveats and limitations (p. 13).

To finish the basic migration - manual steps

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. The console should automatically use the latest version of AWS WAF. To verify this, in the navigation pane, check that you can see the option **Switch to AWS WAF Classic**. If you see **Switch to new AWS WAF**, choose that to switch to the latest version.
3. In the navigation pane, choose **Web ACLs**.
4. In the **Web ACLs** page, locate your new web ACL in the list for the Region where you created it. Choose the web ACL's name to bring up the settings for the web ACL.
5. Review all of the settings for the new web ACL against your prior AWS WAF Classic web ACL. By default, logging and protected resource associations are disabled. You enable those when you're ready to switch over.
6. If your AWS WAF Classic web ACL had a rate-based rule with a condition, the condition wasn't brought over in the migration. You can add conditions to the rule in the new web ACL.

   a. In your web ACL settings page, choose the **Rules** tab.
   b. Locate your rate-based rule in the list, select it, and choose **Edit**.
   c. For **Criteria to count request towards rate limit**, select **Only consider requests that match the criteria in a rule statement**, then provide your additional criteria. You can add the criteria using any rule statement that can be nested, including logical statements. For information about your choices, see Rule statements list (p. 61).
7. If your AWS WAF Classic web ACL had a managed rule group, the rule group inclusion wasn't brought over in the migration. You can add managed rule groups to the new web ACL. Review the information about managed rule groups, including the list of AWS Managed Rules that are available with the new version of AWS WAF, at Managed rule groups (p. 30). To add a managed rule group, do the following:

   a. In your web ACL settings page, choose the web ACL **Rules** tab.
   b. Choose **Add rules**, then choose **Add managed rule groups**.
   c. Expand the listing for the vendor of your choice and select the rule groups that you want to add. For AWS Marketplace sellers, you might need to subscribe to the rule groups. For more information about using managed rule groups in your web ACL, see Managed rule groups (p. 30) and Web ACL rule and rule group evaluation (p. 19).

After you finish the basic migration process, we recommend that you review your needs and consider additional options, to be sure that the new configuration is as efficient as possible and that it's using the latest available security options. See Migrating a web ACL: additional considerations (p. 16).

Migrating a web ACL: additional considerations

Review your new web ACL and consider the options available to you in the new AWS WAF to be sure that the configuration is as efficient as possible and that it's using the latest available security options.

Additional AWS Managed Rules

Consider implementing additional AWS Managed Rules in your web ACL to increase the security posture for your application. These are included with AWS WAF at no additional cost. AWS Managed Rules feature the following types of rule groups:

- Baseline rule groups provide general protection against a variety of common threats, such as stopping known bad inputs from making it into your application and preventing admin page access.

- Use-case specific rule groups provide incremental protection for many diverse use cases and environments.
- IP reputation lists provide threat intelligence based on the client's source IP.

For more information, see AWS Managed Rules for AWS WAF (p. 37).

Rule optimization and cleanup

Revisit your old rules and consider optimizing them by rewriting them or removing outdated ones. For example, if in the past, you deployed an AWS CloudFormation template from the technical paper for OWASP Top 10 Web Application Vulnerabilities, Prepare for the OWASP Top 10 Web Application Vulnerabilities Using AWS WAF and Our New White Paper, you should consider replacing that with AWS Managed Rules. While the concept found within the document is still applicable and may assist you in writing your own rules, the rules created by the template have been largely superseded by AWS Managed Rules.

Amazon CloudWatch metrics and alarms

Revisit your Amazon CloudWatch metrics and set up alarms as needed. The migration doesn't carry over CloudWatch alarms and it's possible that your metric names aren't what you want.

Review with your application team

Work with your application team and check your security posture. Find out what fields are parsed frequently by the application and add rules to sanitize the input accordingly. Check for any edge cases and add rules to catch these cases if the application's business logic fails to process them.

Plan the switchover

Plan the timing of the switch with your application team. The switch from the old web ACL association to the new one can cause a brief disruption.

When you are ready to switch over, follow the procedure at Migrating a web ACL: switchover (p. 17).

Migrating a web ACL: switchover

After you've verified your new web ACL settings, you can start to use it in place of your AWS WAF Classic web ACL.

To begin using your new AWS WAF web ACL

1. Associate the AWS WAF web ACL with the resources that you want to protect, following the guidance at Associating or disassociating a web ACL with an AWS resource (p. 26). This automatically disassociates the resources from the old web ACL.
2. Configure logging for the new web ACL, following the guidance at Logging web ACL traffic information (p. 107).
3. (Optional) If your AWS WAF Classic web ACL is no longer associated with any resources, consider removing it entirely from AWS WAF Classic. For information, see Deleting a Web ACL (p. 202).

Managing and using a web access control list (web ACL)

A web access control list (web ACL) gives you fine-grained control over all of the HTTP(S) web requests that your protected resource responds to. You can protect Amazon CloudFront, Amazon API Gateway, Application Load Balancer, and AWS AppSync resources.

You can use criteria like the following to allow or block requests:

- IP address origin of the request
- Country of origin of the request
- String match or regular expression (regex) match in a part of the request
- Size of a particular part of the request
- Detection of malicious SQL code or scripting

You can also test for any combination of these conditions. You 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. You can combine conditions using logical operators.

This criteria is provided inside the rules that you include in your web ACL and in rule groups that you use in the web ACL. It's specified in the rule statement. For a full list of the options, see AWS WAF rule statements (p. 60).

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, either allow or block, for web requests that don't match any of the rules that you specify. For more information, see Deciding on the default action for a web ACL (p. 21).
2. Add any rule groups that you want to use in your web ACL. Managed rule groups usually contain rules that block web requests. For information about rule groups, see Rule groups (p. 30).
3. Specify additional conditions under which you want to allow or block requests in one or more rules. To add more than one, start with `AND` or `OR` rule statements and nest the rules that you want to combine under those. If you want to negate a rule option, nest the rule in a NOT statement. You can optionally use a rate-based rule instead of a regular rule to limit the number of requests from any single IP address that meets the conditions. For information about rules, see AWS WAF rules (p. 52).

If you add more than one rule to a web ACL, AWS WAF evaluates the rules in the order that they're listed for the web ACL. For more information, see Web ACL rule and rule group evaluation (p. 19).

When you create a web ACL, you specify the types of resources that you want to use it with. For information, see Creating a web ACL (p. 22). After you define a web ACL, you can associate it with your resources to begin providing protection for them. For more information, see Associating or disassociating a web ACL with an AWS resource (p. 26).

How AWS resources handle response delays from AWS WAF

On some occasions, AWS WAF might encounter an internal error that delays the response to associated AWS resources about whether to allow or block a request. On those occasions, CloudFront typically allows the request or serves the content, while the Regional services typically deny the request and don't serve the content.

Topics

Web ACL rule and rule group evaluation

The way a web ACL handles a web request depends on the following:

- The ordering of the rules and rule groups
- The action settings on the rules and web ACL
- Any overrides that you place on the rules and rule groups that you add

For a list of the rule action settings, see AWS WAF rule action (p. 53).

You can customize request and response handling in your rule action settings and default web ACL action settings. For information, see Customizing web requests and responses in AWS WAF (p. 89).

Processing order of rules and rule groups in a web ACL

If you add more than one rule or rule group to a web ACL, AWS WAF evaluates each web request against them in the order that you list them in the web ACL. In each rule group, AWS WAF processes the rules in the order in which they're listed in the rule group.

For example, say you have the following rules and rule groups in your web ACL, ordered as shown:

- Rule1
- RuleGroupA
  - RuleA1
  - RuleA2
- Rule2
- RuleGroupB
  - RuleB1
  - RuleB2

AWS WAF would evaluate the rules for this web ACL in the following order:

- Rule1
- RuleGroupA RuleA1
- RuleGroupA RuleA2
- Rule2
- RuleGroupB RuleB1
- RuleGroupB RuleB2

Basic handling of the rule and rule group actions in a web ACL

When you configure your rules and rule groups, you choose between counting, allowing, or blocking matching web requests:

- **Allow and block are terminating actions** – Allow and block actions stop all other processing of the web ACL on the matching web request. If a rule in a web ACL finds a match for a request and the rule action is allow or block, that match determines the final disposition of the web request for the web ACL. AWS WAF doesn't process any other rules in the web ACL that come after the matching one. This is true for rules that you add directly to the web ACL and rules that are inside an added rule group. With the block action, the protected resource doesn't receive or process the web request.

- **Count is a non-terminating action** – When a rule with a count action matches a request, AWS WAF counts the request, then continues processing the rules that follow in the web ACL rule set. If the only rules that match have count action set, AWS WAF applies the web ACL default action setting.

The actions that AWS WAF applies to a web request are affected by the relative position of rules in the web ACL. For example, say that a web request matches a rule that allows requests and matches another rule that counts requests. If the rule that allows requests is listed first, then AWS WAF won't count the request because the request evaluation terminates with the allow action.

In your web ACL, you can override the action settings for rule groups and their rules. For information, see Overriding the actions of a rule group or its rules (p. 20).

Overriding the actions of a rule group or its rules

When you add a rule group to your web ACL, you can alter how it manages your web requests, so that it counts matching requests rather than acting on them. This can be useful for activities like testing and monitoring a rule group's behavior before you use it. Doing this doesn't alter the rule group itself. It only alters how AWS WAF uses the rule group in the context of the web ACL.

Setting the rule actions to count

You can override the actions of the rules inside a managed rule group, setting them to count for some or all of the rules. If a rule's action is configured inside the rule group to block or allow matching requests, this override changes that behavior so that matching requests are only counted.

This allows all of the rules in the rule group to run and generate logs and metrics without affecting how the web request is handled and without terminating the evaluation of the web request.

When AWS WAF evaluates a web request against a rule group whose rules are all set to count by the web ACL, if the request matches one of the rules, AWS WAF processes the match normally and then continues evaluating the subsequent rules in the rule group. The count action is non-terminating and doesn't prompt a match response from the rule group in the web ACL.

You can use this option to test a rule group or inspect it for false positives. It allows AWS WAF to evaluate web requests against all rules in the rule group and report all matches that it finds to your configured samples, metrics, and logs.

This can help you troubleshoot false positives, which is when a rule group blocks traffic that you aren't expecting it to block. If you identify a rule within the rule group that's blocking requests that you want to allow through, you can leave that rule in count mode, to exclude it from acting on your requests.

If you want to test a rule before you start using it to allow or block requests, configure AWS WAF to count the web requests that match the conditions in the rule. If you have metrics enabled, you'll receive `COUNT` metrics for the rule whose action is set to count. For more information, see Testing web ACLs (p. 27).

For information about how to use this option, see Setting rule actions to count in a rule group (p. 24).

Overriding the resulting rule group's action to count

You can override the action that the rule group returns to count.

When you override the rule group action to count, the allow and block actions affect the processing of the rules inside the rule group as described at Basic handling of the rule and rule group actions in a web ACL (p. 19), but then at the web ACL level, the action override makes the rule group resulting action a count action, and the web ACL doesn't discontinue processing. It processes the rest of the rules in the web ACL as if the result from the rule group were a match with count action.

With this option, the rules in the rule group run as specified by the rule group configuration. The first rule that matches a web request and that has a rule action of allow or block terminates the rule group evaluation and returns the allow or block action for the rule group. At this point, the web ACL override changes the returned rule group action to count and then continues processing any other rules and rule groups that are configured in the web ACL. So, processing inside the rule group works according to the rule group's rule configurations, up to the first rule that matches and has a termination action setting.

If the rules in the rule group find no match for the web request, this setting has no effect.

For information about how to use this option, see Overriding a rule group's action to count (p. 25).

Deciding on the default action for a web ACL

When you create and configure a web ACL, you set the web ACL default action, which determines how AWS WAF handles web requests that don't match any rules in the web ACL. The default action must be a terminating action:

- **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. Then, when you add rules to your web ACL, add rules that identify and block the specific requests that you want to block. With this action, you can insert custom headers into the request before forwarding it to the protected resource.
- **Block** – If you want to prevent most 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. Then when you add rules to your web ACL, add rules that identify and allow the specific requests that you want to allow in. By default, for the block action, the AWS resource responds with an HTTP `403 (Forbidden)` status code, but you can customize the response.

For information about customizing requests and responses, see Customizing web requests and responses in AWS WAF (p. 89).

Your configuration of your own rules and rule groups depends in part on whether you want to allow or block most web requests. For example, if you want to *allow* most requests, you would set the web ACL default action to allow, and then add rules that identify 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 originate from countries that either you don't do business in or are the frequent source of attacks
- Requests that include fake values in the `User-agent` header
- Requests that appear to include malicious SQL code

Managed rule groups usually use the block action. For information about managed rule groups, see Managed rule groups (p. 30).

Working with web ACLs

When you make changes to web ACLs or web ACL components, like rules and rule groups, AWS WAF propagates the changes everywhere that the web ACL and its components are stored and used. Your changes are applied within seconds, but there might be a brief period of inconsistency when the changes have arrived in some places and not in others. So, for example, if you add an IP address to an IP set that's referenced by a blocking rule in a web ACL, the new address might briefly be blocked in one area while still allowed in another. This temporary inconsistency can occur when you first associate a web ACL with

an AWS resource and when you change a web ACL that is already associated with a resource. Generally, any inconsistencies of this type last only a few seconds.

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/wafv2/.
2. Choose **Web ACLs** in the navigation pane, and then choose **Create web ACL**.
3. For **Name**, enter the name that you want to use to identify this web ACL.

   Note
   You can't change the name after you create the web ACL.
4. (Optional) For **Description - optional**, enter a longer description for the web ACL if you want to.
5. For **CloudWatch metric name**, change the default name if applicable. Follow the guidance on the console for valid characters. The name can't contain special characters, white space, or metric names reserved for AWS WAF, including "All" and "Default_Action."

   Note
   You can't change the CloudWatch metric name after you create the web ACL.
6. For **Resource type**, choose the category of AWS resource that you want to associate with this web ACL. For more information, see Associating or disassociating a web ACL with an AWS resource (p. 26).
7. For **Region**, if you've chosen a Regional resource type, choose the Region where you want AWS WAF to store the web ACL.

   You only need to choose this option for Regional resource types. For CloudFront distributions, the Region is hard-coded to the US East (N. Virginia) Region, `us-east-1`, for Global (CloudFront) applications.
8. (Optional) For **Associated AWS resources - optional**, choose **Add AWS resources**. In the dialog box, choose the resources that you want to associate, and then choose **Add**. AWS WAF returns you to the **Describe web ACL and associated AWS resources** page.
9. Choose **Next**.
10. (Optional) If you want to add managed rule groups, on the **Add rules and rule groups** page, choose **Add rules**, and then choose **Add managed rule groups**. Do the following for each managed rule group that you want to add:

    a. On the **Add managed rule groups** page, expand the listing for AWS managed rule groups or for the AWS Marketplace seller of your choice.

    b. For the rule group that you want to add, turn on the **Add to web ACL** toggle in the **Action** column.

       If you want to set the actions for all rules in the rule group to count only, choose **Edit**, then turn on the **Set all rules actions to count** toggle, and choose **Save rule**. For information about this option, see Overriding the actions of a rule group or its rules (p. 20).

Choose **Add rules** to finish adding managed rules and return to the **Add rules and rule groups** page.

11. (Optional) If you want to add your own rule group, on the **Add rules and rule groups** page, choose **Add rules**, and then choose **Add my own rules and rule groups**. Do the following for each rule group that you want to add:

    a. On the **Add my own rules and rule groups** page, choose **Rule group**.

    b. Choose your rule group from the list, and then choose **Add rule**.

12. (Optional) If you want to add your own rule, on the **Add rules and rule groups** page, choose **Add rules**, **Add my own rules and rule groups**, **Rule builder**, then **Rule visual editor**.

    Note
    The console **Rule visual editor** supports one level of nesting. For example, you can use a single logical `AND` or `OR` statement and nest one level of other statements inside it, but you can't nest logical statements within logical statements. To manage more complex rule statements, use the **Rule JSON editor**. For information about all options for rules, see AWS WAF rules (p. 52).
    This procedure covers the **Rule visual editor**.

    a. For **Name**, enter the name that you want to use to identify this rule.

    b. Enter your rule definition, according to your needs. You can combine rules inside logical `AND` and `OR` rule statements. The wizard guides you through the options for each rule, according to context. For information about your rules options, see AWS WAF rules (p. 52).

    c. For **Action**, select the action you want the rule to take when it matches a web request. For information on your choices, see AWS WAF rule action (p. 53) and Web ACL rule and rule group evaluation (p. 19).

    If you want to customize the request or response, choose the options for that and fill in the details of your customization. For more information, see Customizing web requests and responses in AWS WAF (p. 89).

    If you want to have your rule add labels to matching web requests, choose the options for that and fill in your label details. For more information, see AWS WAF labels on web requests (p. 54).

    d. Choose **Add 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 any of the rules in the web ACL. For more information, see Deciding on the default action for a web ACL (p. 21).

    If you want to customize the default action, choose the options for that and fill in the details of your customization. For more information, see Customizing web requests and responses in AWS WAF (p. 89).

14. Choose **Next**.

15. In the **Set rule priority** page, select and move your rules and rule groups to the order that you want AWS WAF to process them. For more information, see Web ACL rule and rule group evaluation (p. 19).

16. Choose **Next**.

17. In the **Configure metrics** page, update your metrics and sampling options as needed. You can combine metrics from multiple sources by providing the same **CloudWatch metric name** for them.

18. Choose **Next**.

19. In the **Review and create web ACL** page, check over your definitions. If you want to change any area, choose **Edit** for the area. This returns you to the page in the web ACL wizard. Make any changes, then choose **Next** through the pages until you come back to the **Review and create web ACL** page.

20. Choose **Create web ACL**. Your new web ACL is listed in the **Web ACLs** page.

Editing a web ACL

To add or remove rules from a web ACL or change the default action, access the web ACL using 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/wafv2/.
2. In the navigation pane, choose **Web ACLs**.
3. Choose the name of the web ACL that you want to edit. The console takes you to the web ACL's description, where you can edit it.

   Note
   Web ACLs that are managed by AWS Firewall Manager have names that start with `FMManagedWebACLV2-`. The Firewall Manager administrator manages these in Firewall Manager AWS WAF policies. These web ACLs might contain rule group sets that are designated to run first and last in the web ACL, on either side of any rules or rule groups that you add and manage. The first and last rule groups have names that start with `PREFMManaged-` and `POSTFMManaged-`, respectively. For more information about these policies, see AWS WAF policies (p. 284).
4. Page through the web ACL definitions, and make your changes. This is the same as the procedure that you use to create the web ACL in Creating a web ACL (p. 22), but with some fields not modifiable. For example, you can't change the name of a web ACL, and for web ACLs that are managed by Firewall Manager, you can't change any first and last rule group specifications.

Managing rule group behavior in a web ACL

This section describes your options for modifying how you use a rule group in your web ACL. This information applies to all rule group types. After you add a rule group to a web ACL, you can override the actions of the individual rules in the rule group to count. You can also override the rule group's resulting action to count, which has no effect on how the rules are evaluated inside the rule group.

For information about these options, see Overriding the actions of a rule group or its rules (p. 20).

Setting rule actions to count in a rule group

For each rule group in a web ACL, you can override the contained rule's actions to count for some or all of the rules. Rules that you alter like this are described as being *excluded rules* in the rule group. If you have metrics enabled, you receive `COUNT` metrics for each excluded rule. This change alters how the rules in the rule group are evaluated.

To set rule actions to count in a rule group

1. After you've added your rule group to your web ACL, edit the web ACL.
2. In the web ACL page **Rules** tab, select the rule group, then choose **Edit**.
3. In the **Rules** section for the rule group, do one of the following:

   - (Option) Turn on the **Set all rule actions to count** toggle.
   - (Option) For each rule that you want to set to count, turn on the **Rule action Count** toggle.
4. Choose **Save rule**.

The following example JSON listing shows a rule group declaration inside a web ACL that sets the rule actions to count for the rules `CategoryVerifiedSearchEngine` and `CategoryVerifiedSocialMedia`. Through the API, in order to set all rule actions to count when you

add a rule group to a web ACL, you list them all by name in `ExcludedRules` specification inside the rule group reference statement, as shown here.

```
{
    "Name": "AWS-AWSBotControl-Example",
   "Priority": 5,
   "Statement": {
    "ManagedRuleGroupStatement": {
        "VendorName": "AWS",
        "Name": "AWSManagedRulesBotControlRuleSet",
        "ExcludedRules": [
            {
                "Name": "CategoryVerifiedSearchEngine"
            },
            {
                "Name": "CategoryVerifiedSocialMedia"
            }
        ]
    },
   "VisibilityConfig": {
       "SampledRequestsEnabled": true,
       "CloudWatchMetricsEnabled": true,
       "MetricName": "AWS-AWSBotControl-Example"
   }
}
```

Overriding a rule group's action to count

You can override the action that a rule group returns to count, without altering how the rules in the rule group are configured or evaluated.

To override the rule group's resulting action

1. After you've added your rule group to your web ACL, edit the web ACL.
2. In the web ACL page **Rules** tab, select the rule group, then choose **Edit**.
3. Enable the option **Override rule group action**.
4. Choose **Save rule**.

The following example JSON listing shows a rule group declaration inside a web ACL where the web ACL is configured to override the rule group action to count. The override settings are in bold.

```
{
   "Name": "AWS-AWSBotControl-Example",
   "Priority": 5,
   "Statement": {
    "ManagedRuleGroupStatement": {
     "VendorName": "AWS",
     "Name": "AWSManagedRulesBotControlRuleSet"
     }
   },
    "OverrideAction": {
       "Count": {}
    },
   "VisibilityConfig": {
        "SampledRequestsEnabled": true,
        "CloudWatchMetricsEnabled": true,
        "MetricName": "AWS-AWSBotControl-Example"
   }
}
```

Associating or disassociating a web ACL with an AWS resource

You can use AWS WAF to associate a web ACL with an Amazon API Gateway REST API, an Application Load Balancer, or an AWS AppSync GraphQL API.

You can associate a web ACL with a CloudFront distribution when you create or update the distribution itself. For information, see Using AWS WAF to Control Access to Your Content in the *Amazon CloudFront Developer Guide*.

You can only associate a web ACL to an Application Load Balancer within AWS Regions. For example, you can't associate a web ACL to an Application Load Balancer that is on AWS Outposts.

Restrictions on multiple associations

You can associate a single web ACL with one or more AWS resources, according to the following restrictions:

- You can associate each AWS resource with only one web ACL. The relationship between web ACL and AWS resources is one-to-many.
- You can associate a web ACL with one or more CloudFront distributions. You can't associate a web ACL that you've associated with a CloudFront distribution with any other AWS resource type.

To associate a web ACL with an AWS resource

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **Web ACLs**.
3. Choose the web ACL that you want to associate with a resource.
4. On the **Associated AWS resources** tab, choose **Add AWS resources**.
5. When prompted, choose your resource that you want to associate this web ACL with. If you choose an Amazon API Gateway REST API, an Application Load Balancer, or an AWS AppSync GraphQL API, specify a Region.
6. Choose **Add**.

To disassociate a web ACL from an AWS resource

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **Web ACLs**.
3. Choose the web ACL that you want to disassociate from your resource.
4. On the **Associated AWS resources** tab, deselect the resources that you want to disassociate this web ACL from.
5. Choose **Save**.

Deleting a web ACL

To delete a web ACL, you first disassociate all AWS resources 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/wafv2/.
2. In the navigation pane, choose **Web ACLs**.

3. Select the name of the web ACL that you want to delete. The console takes you to the web ACL's description, where you can edit it.
4. On the **Associated AWS resources** tab, select all resources, and then choose **Remove** to disassociate the web ACL from all resources.
5. In the navigation pane, choose **Web ACLs**.
6. Select the radio button next to the web ACL that you are deleting, 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. For information about how to do this for a rule group in a web ACL, see Setting rule actions to count in a rule group (p. 24).
- Set the default action for the web ACL to allow requests.

In this configuration, AWS WAF inspects each web request based on the match statement in the first rule. If the web request matches a rule, AWS WAF increments a counter for that rule. Then AWS WAF inspects the web request based on the match statement in the next rule. If the web request matches the rule, AWS WAF increments a counter for the rule. This continues until AWS WAF has inspected the request against the match statements 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 one or more AWS resources (an Amazon CloudFront distribution, an Amazon API Gateway REST API, an Application Load Balancer, or an AWS AppSync GraphQL API) you can view the resulting counts in an Amazon CloudWatch graph. For each rule in a web ACL and for all the requests that an associated resource forwards to AWS WAF for a web ACL, CloudWatch lets you do the following:

- 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 Management 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 **AWS/WAFV2**.
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 recently associated a web ACL with an AWS resource, 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 resource with a web ACL, the CloudWatch data will include requests for all of them.
   - 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 an associated AWS resource has forwarded to AWS WAF. For more information, see Viewing a sample of web requests (p. 28).
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 Managing and using a web access control list (web ACL) (p. 17).

   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. 24).

Viewing a sample of web requests

In the AWS WAF console, if you have request sampling enabled, you can view a sample of the requests that an associated resource 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 protected resources that have received requests for your content in the previous 15 minutes.

To view a sample of the web requests that an associated resource 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/wafv2/.

2. In the navigation pane, choose **Web ACLs**
3. Choose the web ACL for which you want to view requests.
4. In the **Overview** 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 Amazon CloudFront, Amazon API Gateway, Application Load Balancer, or AWS AppSync.
5. 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.
6. To refresh the list of sample requests, choose **Get new samples**.

Rule groups

A *rule group* is a reusable set of rules that you can add to a web ACL. For more information about web ACLs, see Managing and using a web access control list (web ACL) (p. 17).

Rule groups fall into two main categories:

- Managed rule groups, which AWS Managed Rules and AWS Marketplace sellers create and maintain for you
- Your own rule groups, which you create and maintain

Differences between rule groups and web ACLs

Rule groups and web ACLs both contain rules, which are defined in the same manner in both places. Rule groups differ from web ACLs in the following ways:

- Rule groups can't contain rule group reference statements.
- You can reuse a single rule group in multiple web ACLs by adding a rule group reference statement to each web ACL. You can't reuse a web ACL.
- Rule groups don't have default actions. In a web ACL, you set a default action for each rule or rule group that you include. Each individual rule inside a rule group or web ACL has an action defined.
- You don't directly associate a rule group with an AWS resource. To protect resources using a rule group, you use the rule group in a web ACL.
- Web ACLs have a system-defined maximum capacity of 1,500 web ACL capacity units (WCUs). Every rule group has a WCU setting that must be set at creation. You can use this setting to calculate the additional capacity requirements that using a rule group would add to your web ACL.

For information about rules, see AWS WAF rules (p. 52).

This section describes the types of managed rule groups that are available to you and provides guidance for creating and managing your own rule groups, if you choose to do so.

Topics

Managed rule groups

Managed rule groups are collections of predefined, ready-to-use rules that AWS and AWS Marketplace sellers write and maintain for you:

- *AWS Managed Rules rule groups* are mostly available for free to AWS WAF customers. The AWS WAF Bot Control rule group has additional fees. For more information, see AWS WAF Pricing.
- *AWS Marketplace managed rule groups* are available by subscription through AWS Marketplace.

Some managed rule groups are designed to help protect specific types of web applications like WordPress, Joomla, or PHP. Others offer broad protection against known threats or common web application vulnerabilities, such as those listed in the OWASP Top 10. If you're subject to regulatory compliance like PCI or HIPAA, you might be able to use managed rule groups to satisfy web application firewall requirements.

Automatic updates

Keeping up to date on the constantly changing threat landscape can be time consuming and expensive. Managed rule groups can save you time when you implement and use AWS WAF. AWS and AWS Marketplace sellers automatically update managed rule groups and provide new versions of rule groups when new vulnerabilities and threats emerge.

AWS and many of the AWS Marketplace sellers are notified of new vulnerabilities before public disclosure. AWS WAF can update their rule groups and deploy them to you even before a new threat is widely known. Many also have threat research teams to investigate and analyze the most recent threats in order to write the most relevant rules.

Restricted access to rules in a managed rule group

Each managed rule group provides a comprehensive description of the types of attacks and vulnerabilities that it's designed to protect against. To protect the intellectual property of the rule group providers, you can't view details for the individual rules within a rule group. This restriction also helps to keep malicious users from designing threats that specifically circumvent published rules.

Topics

Version management with managed rule groups

A managed rule group provider updates a rule group's options and capabilities in new versions of the rule group. Usually, a specific version of a managed rule group is static. Occasionally, a provider might need to update some or all of their existing versions of a managed rule group, for example, to respond to an emerging security threat.

When you add a managed rule group to your web ACL, if the rule group supports versioning, you can choose to let the provider manage which version you use or you can manage the version setting yourself.

Topics

Version lifecycle for managed rule groups

Providers handle the following lifecycle stages of a managed rule group version:

- **Release and updates** – A managed rule group provider announces upcoming and new versions of a managed rule group through notifications to an Amazon Simple Notification Service (Amazon SNS) topic. Providers might also use the topic to communicate other important information about the rule group, such as urgent required updates.

  You can subscribe to the rule group's topic and configure how you want to receive notifications. For more information see Getting notified of new versions and updates (p. 35).
- **Expiration scheduling** – A managed rule group provider schedules older versions of a rule group for expiration. A version that's scheduled to expire cannot be added to your web ACL rules. After expiration is scheduled for a version, AWS WAF tracks the expiration with a countdown metric in Amazon CloudWatch.

  You can set an alarm on the metric in CloudWatch to track the expiration of a version that you're using. This gives you time to test a new version and move off of the expiring one before the countdown completes. For more information, see Tracking version expiration (p. 36).

- **Version expiration** – When a version expires, the rule group provider determines how to manage the expiration for web ACLs that are still using the expired version:
  - For AWS Managed Rules rule groups, AWS WAF moves any web ACL that's using the expired version to the rule group's default version.
  - For AWS Marketplace rule groups, the provider determines how to handle the expiration. Ask your managed rule group provider for information.

Best practices for handling managed rule group versions

Follow this versioning best practice guidance when you use a managed rule group.

When you use a managed rule group to your web ACL, you can choose to use a specific, static version of the rule group, or you can choose to use the default version:

- **Default version** – If you choose the default version, AWS WAF always uses the version that's currently recommended by the provider. When the provider updates their recommended version, AWS WAF automatically updates the version for the rule group in your web ACL.

  When you use the default version of a managed rule group, do the following as best practice:

  - **Subscribe to notifications** – Subscribe to notifications for changes to the rule group and keep an eye on those. Most providers provide advanced notification of version changes to so that you can make plans to check the effects of a new version when it's released. If you're managing the rule group at the rule level, you might need to adjust your settings for the new version, to accommodate new rules and other rule-level changes. You'll also receive notification when a new version is released. For more information see Getting notified of new versions and updates (p. 35).
  - **Review the effects of new versions** – When a rule group version changes, review the effects of the new version on your web request monitoring and management. The new version might have new rules to review. Look for false positives or other unexpected behavior, in case you need to modify how you use the rule group. You can set rules to count, for example, to stop them from blocking traffic while you figure out how you want to handle the new behavior.
- **Static version** – If you choose to use a static version, you must manually update the version setting when you're ready to adopt a new version of the rule group.

  When you use a static version of a managed rule group, do the following as best practice:

  - **Keep your version up to date** – Keep your managed rule group as close as you can to the latest version. The latest version of any rule group contains the provider's best approach to protecting your resources. When a new version is released, test it, adjust settings as needed, and implement it in a timely manner.
  - **Subscribe to notifications** – Subscribe to notifications for changes to the rule group, so you know when your provider releases new versions. Most providers give advanced notification of version changes. Additionally, your provider might need to update the rule group version you're using to close a security loophole or for other urgent reasons. You'll know what's happening if you're subscribed to the provider's notifications. For more information, see Getting notified of new versions and updates (p. 35).
  - **Avoid version expiration** – Don't allow a version to expire while you're using it. Provider handling of expired versions can vary and might include forcing an upgrade to an available version or other changes that can have unexpected consequences. Track the AWS WAF expiry metric and set an alarm that gives you a sufficient number of days to successfully upgrade to a supported version. For more information, see Tracking version expiration (p. 36).

Working with managed rule groups

This section provides guidance for accessing and managing your managed rule groups.

When you add a managed rule group to your web ACL, you can choose the same configuration options as you can your own rule groups, plus additional settings.

Through the console, you access managed rule group information during the process of adding and editing the rules in your web ACLs. Through the APIs and the command line interface (CLI), you can directly request managed rule group information.

When you use a managed rule group in your web ACL, you can edit the following settings:

- **Version** – This is available only if the rule group is versioned. For more information, see Version management with managed rule groups (p. 31).
- **Set rule actions to Count** – You can set the actions for rules in the rule group to Count. This is useful for testing a rule group before using it to manage your web requests. For more information, see Setting the rule actions to count (p. 20).
- **Scope-down statement** – You can add a scope-down statement, to filter out web requests that you don't want to evaluate with the rule group. For more information, see Scope-down statements (p. 81).
- **Override rule group action** – You can override the action that results from the rule group evaluation, and set it to Count only. This option isn't commonly used. It doesn't alter how AWS WAF evaluates the rules in the rule group. For more information, see Overriding the resulting rule group's action to count (p. 20).

To edit the managed rule group settings in your web ACL

- **Console**
  - (Option) When you add the managed rules group to your web ACL, you can choose **Edit** to view and edit the settings.
  - (Option) After you've added the managed rule group into your web ACL, from the **Web ACLs** page, choose the web ACL you just created. This takes you to the web ACL edit page.
    - Choose **Rules**.
    - Select the rule group, then choose **Edit** to view and edit the settings.
- **APIs and CLI** – Outside of the console, you can manage the managed rule group settings when you create and update the web ACL.

Retrieving the list of managed rule groups

The managed rule groups that are available for you to use in your web ACLs are the following:

- All AWS Managed Rules rule groups.
- The AWS Marketplace rule groups that you have subscribed to.

  Note
  For information about subscribing to AWS Marketplace rule groups, see AWS Marketplace managed rule groups (p. 49).

When you retrieve the list of managed rule groups, the list you get back depends on the interface that you're using:

- **Console** – Through the console, you can see all managed rule groups, including the AWS Marketplace rule groups that you haven't subscribed to yet. For the ones that you haven't subscribed to yet, the interface provides links that you can follow to subscribe.
- **APIs and CLI** – Outside of the console, your request returns only the rule groups that are available for you to use.

To retrieve the list of managed rule groups

- **Console** – During the process of creating a web ACL, on the **Add rules and rule groups** page, choose **Add managed rule groups**. At the top level, the provider names are listed. Expand each provider listing to see the list of managed rule groups. For versioned rule groups, the information shown at this level is for the default version. When you add a managed rule group to your web ACL, the console lists it based on the naming scheme `<Vendor Name>-<Managed Rule Group Name>`.
- **API** –
  - `ListAvailableManagedRuleGroups`
- **CLI** –
  - `aws wafv2 list-available-managed-rule-groups`

Retrieving the rules in a managed rule group

You can retrieve a list of the rules in a managed rule group. The API and CLI calls return the rules specifications that you can reference in the JSON model or through AWS CloudFormation.

To retrieve the list of rules in a managed rule group

- **Console**
  - (Option) When you add the managed rules group to your web ACL, you can choose **Edit** to view the rules.
  - (Option) After you've added the managed rule group into your web ACL, from the **Web ACLs** page, choose the web ACL you just created. This takes you to the web ACL edit page.
    - Choose **Rules**.
    - Select the rule group you want to see a rules list for, then choose **Edit**. AWS WAF shows the list of rules in the rule group.
- **API** – `DescribeManagedRuleGroup`
- **CLI** – `aws wafv2 describe-managed-rule-group --scope REGIONAL --vendor-name <vendor> --name <managedrule_name>`

Retrieving the available versions for a managed rule group

The available versions of a managed rule group are versions that haven't yet been scheduled to expire.

To retrieve a list of the available versions of a managed rule group

- **Console**
  - (Option) When you add the managed rule group to your web ACL, choose **Edit** to see the rule group's information. Expand the **Version** dropdown to see the list of available versions.
  - (Option) After you've added the managed rule group into your web ACL, choose **Edit** on the web ACL, and then select and edit the rule group rule. Expand the **Version** dropdown to see the list of available versions.
- **API** –
  - `ListAvailableManagedRuleGroupVersions`
- **CLI** –
  - `aws wafv2 list-available-managed-rule-group-versions --scope REGIONAL --vendor-name <vendor> --name <managedrule_name>`

Getting notified of new versions and updates to a managed rule group

You can subscribe to notifications for updates to a managed rule group. Providers use notifications to announce rule group changes, like upcoming new versions and urgent security updates.

How to subscribe

To subscribe to notifications for a rule group, you create an Amazon Simple Notification Service (Amazon SNS) subscription for the rule group's Amazon SNS topic ARN. For information about how to subscribe, see the Amazon Simple Notification Service Developer Guide.

Where to find the Amazon SNS topic ARN for a managed rule group

- **Console**
  - (Option) When you add the managed rule group to your web ACL, choose **Edit** to see the rule group's information, which includes the rule group's Amazon SNS topic ARN.
  - (Option) After you've added the managed rule group into your web ACL, choose **Edit** on the web ACL, and then select and edit the rule group rule to see the rule group's Amazon SNS topic ARN.
- **API** – `DescribeManagedRuleGroup`
- **CLI** – `aws wafv2 describe-managed-rule-group --scope REGIONAL --vendor-name <vendor> --name <managedrule_name>`

The notification format for AWS Managed Rules

The Amazon SNS notifications for AWS Managed Rules rule groups always contain the fields `Subject`, `Message`, and `MessageAttributes`. Other fields are included according to the type of message and which managed rule group the notification is for.

The following shows an example notification listing for the `AWSManagedRulesCommonRuleSet`.

```
{
    "Type": "Notification",
    "MessageId": "4286b830-a463-5e61-bd15-e1ae72303868",
    "TopicArn": "arn:aws:sns:us-west-2:123456789012:MyTopic",
    "Subject": "New version available for rule group AWSManagedRulesCommonRuleSet",
    "Message": "Welcome to AWSManagedRulesCommonRuleSet version 1.5! We've updated
 the regex specification in this version to improve protection coverage, adding
 protections against insecure deserialization. For details about this change, see http://
updatedPublicDocs.html. Look for more exciting updates in the future! ",
    "Timestamp": "2021-08-24T11:12:19.810Z",
    "SignatureVersion": "1",
    "Signature": "EXAMPLEHXgJm...",
    "SigningCertURL": "https://sns.us-west-2.amazonaws.com/SimpleNotificationService-
f3ecfb7224c7233fe7bb5f59f96de52f.pem",
    "SubscribeURL": "https://sns.us-west-2.amazonaws.com/?
Action=ConfirmSubscription&TopicArn=arn:aws:sns:us-
west-2:123456789012:MyTopic&Token=2336412f37...",
    "MessageAttributes": {
        "major_version": {
            "Type": "String",
            "Value": "v1"
        },
        "managed_rule_group": {
            "Type": "String",
            "Value": "AWSManagedRulesCommonRuleSet"
        }
    }
}
```

For general information about Amazon SNS notification formats and how to filter the notifications that you receive, see https://docs.aws.amazon.com/sns/latest/dg/sns-message-and-json-formats.html and https://docs.aws.amazon.com/sns/latest/dg/sns-subscription-filter-policies.html in the Amazon Simple Notification Service Developer Guide.

Tracking a rule group's version expiration

If you use a specific version of a rule group, make sure that you don't keep using a version past its expiration date.

> **Tip**
> If you track upcoming changes for the managed rule group through Amazon SNS, you'll receive notifications about new, recommended versions. If you regularly test and move to a newer version, you'll stay ahead of any expiration activities on the older versions. You'll also benefit from the best current protections from the rule group. For information about notifications, see Getting notified of new versions and updates (p. 35).

If a version that you're using is expired, AWS WAF blocks modifications to the web ACL where you're using the rule group. The block remains until you update the rule group to an available version or remove it from your web ACL.

Expiration handling for a managed rule group depends on the rule group provider. For AWS Managed Rules rule groups, the version is automatically changed to the rule group's default version. For AWS Marketplace rule groups, ask the provider how they handle expiration.

To monitor expiration scheduling for a managed rule group, track the Amazon CloudWatch expiry metrics from AWS WAF:

- Metric name: `DaysToExpiry`
- Metric dimensions: `Region`, `ManagedRuleGroup`, `Vendor`, and `Version`

Locate the metric for your managed rule group in Amazon CloudWatch and set an alarm on it so that you're notified in time to switch to a newer version of your rule group. For information about using Amazon CloudWatch metrics and configuring alarms, see the Amazon CloudWatch User Guide.

When the provider creates a new version of the rule group, it sets the version's forecasted lifetime. While the version isn't scheduled to expire, the metric value is set to the forecasted lifetime setting, and in CloudWatch, you'll see a flat value for the metric. After the provider schedules the metric to expire, the metric value diminishes each day until it reaches zero on the day of expiration.

If you have a managed rule group in your web ACL that's evaluating traffic, you will get a metric for it. The metric isn't available for unused rule groups.

Example managed rule group configurations in JSON and YAML

The API and CLI calls return a list of all rules in the managed rule group that you can reference in the JSON model or through AWS CloudFormation.

JSON

You can reference and modify managed rule groups within a rule statement using JSON. The following listing shows the AWS Managed Rules rule group, `AWSManagedRulesCommonRuleSet`, in JSON format. The `ExcludedRules` specification lists rules whose actions are overridden to count only.

```
{
    "Name": "AWS-AWSManagedRulesCommonRuleSet",
    "Priority": 0,
    "Statement": {
      "ManagedRuleGroupStatement": {
        "VendorName": "AWS",
        "Name": "AWSManagedRulesCommonRuleSet",
```

```
        "ExcludedRules": [
          {
            "Name": "NoUserAgent_HEADER"
          }
        ]
      }
    },
    "OverrideAction": {
      "None": {}
    },
    "VisibilityConfig": {
      "SampledRequestsEnabled": true,
      "CloudWatchMetricsEnabled": true,
      "MetricName": "AWS-AWSManagedRulesCommonRuleSet"
    }
}
```

YAML

You can reference and modify managed rule groups within a rule statement using the AWS CloudFormation YAML template. The following listing shows the AWS Managed Rules rule group, `AWSManagedRulesCommonRuleSet`, in AWS CloudFormation template. The `ExcludedRules` specification lists rules whose actions are overridden to count only.

```
Description: WebACL With AMR
Resources:
  WebACLWithAMR:
    Type: AWS::WAFv2::WebACL
    Properties:
      Name: WebACLWithAMR
      Scope: REGIONAL
      Description: This is a demo
      DefaultAction:
        Block: {}
      VisibilityConfig:
        SampledRequestsEnabled: true
        CloudWatchMetricsEnabled: true
        MetricName: MetricForWebACLWithAMR
      Tags:
        - Key: sampleapple
          Value: sampleorange
      Rules:
        - Name: AWS-AWSManagedRulesCommonRuleSet
          Priority: 0
          OverrideAction:
            None: {}
          VisibilityConfig:
            SampledRequestsEnabled: true
            CloudWatchMetricsEnabled: true
            MetricName: MetricForAMRCRS
          Statement:
            ManagedRuleGroupStatement:
              VendorName: AWS
              Name: AWSManagedRulesCommonRuleSet
              ExcludedRules:
                - Name: NoUserAgent_HEADER
```

AWS Managed Rules for AWS WAF

AWS Managed Rules for AWS WAF is a managed service that provides protection against common application vulnerabilities or other unwanted traffic, without having to write your own rules. You have the option of selecting one or more rule groups from AWS Managed Rules for each web ACL, up to the

allowed maximum web ACL capacity unit (WCU) limit. You can choose whether to count (monitor) or block requests that are matched by the managed rules.

As a best practice, before using a rule group in production, test it in a non-production environment, with the action override set to count. Evaluate the rule group using Amazon CloudWatch metrics combined with AWS WAF sampled requests or AWS WAF logs. When you're satisfied that the rule group does what you want, remove the override on the group.

Mitigating False Positive Scenarios

If you are encountering false-positive scenarios with AWS Managed Rules rule groups, perform the following steps:

1. In the web ACL configuration, override the actions in the rules of the rule groups, putting them into count (alert) mode. This stops them from blocking legitimate traffic.
2. Use either AWS WAF sampled requests or AWS WAF logs to identify which AWS Managed Rules rule group is triggering the false positive. You can identify the AWS Managed Rules rule group by looking at the `ruleGroupId` field in the log or the `RuleWithinRuleGroup` in the sampled request. The rule name follows this pattern: `AWS#<AMR RuleGroup Name>#<AMR Rule Name>`.
3. On the AWS WAF console, edit the web ACL, locate the AWS Managed Rules rule group that you've identified, remove your count override for the rules that aren't causing the false positive, and leave the rule that is causing the false positive in count mode.

For more information about a rule in an AWS Managed Rules rule group, contact the AWS Support Center.

AWS Managed Rules rule groups list

This section describes the AWS Managed Rules rule groups that are currently available. You see these on the console when you add a managed rule group to your web ACL. Through the API, you can retrieve this list along with the AWS Marketplace managed rule groups that you're subscribed to by calling `ListAvailableManagedRuleGroups`.

Baseline rule groups

Baseline managed rule groups provide general protection against a wide variety of common threats. Choose one or more of these rule groups to establish baseline protection for your resources.

Core rule set (CRS)

VendorName: `AWS`, Name: `AWSManagedRulesCommonRuleSet`, WCU: 700

The Core rule set (CRS) rule group contains rules that are generally applicable to web applications. This provides protection against exploitation of a wide range of vulnerabilities, including high risk and commonly occurring vulnerabilities described in OWASP publications such as OWASP Top 10. Consider using this rule group for any AWS WAF use case.

Rule name	Description
`NoUserAgent_HEADER`	Blocks requests with no HTTP `User-Agent` header.
`UserAgent_BadBots_HEADER`	Inspects for the presence of common `User-Agent` header values indicating the request to be a bad bot. Example patterns include `nessus`, and `nmap`. For bot management, see also AWS WAF Bot Control rule group (p. 45).

Rule name	Description
`SizeRestrictions_QUERYSTRING`	Verifies that the URI query string length is at most 2,048 bytes.
`SizeRestrictions_Cookie_HEADER`	Verifies that the cookie header length is at most 10,240 bytes.
`SizeRestrictions_BODY`	Verifies that the request body size is at most 10,240 bytes.
`SizeRestrictions_URIPATH`	Verifies that the URI path length is at most 1,024 bytes.
`EC2MetaDataSSRF_BODY`	Inspects for attempts to exfiltrate Amazon EC2 metadata from the request body.
`EC2MetaDataSSRF_COOKIE`	Inspects for attempts to exfiltrate Amazon EC2 metadata from the request cookie.
`EC2MetaDataSSRF_URIPATH`	Inspects for attempts to exfiltrate Amazon EC2 metadata from the request URI path.
`EC2MetaDataSSRF_QUERYARGUMENTS`	Inspects for attempts to exfiltrate Amazon EC2 metadata from the request query arguments.
`GenericLFI_QUERYARGUMENTS`	Inspects for the presence of Local File Inclusion (LFI) exploits in the query arguments. Examples include path traversal attempts using techniques like `../../`.
`GenericLFI_URIPATH`	Inspects for the presence of Local File Inclusion (LFI) exploits in the URI path. Examples include path traversal attempts using techniques like `../../`.
`GenericLFI_BODY`	Inspects for the presence of Local File Inclusion (LFI) exploits in the request body. Examples include path traversal attempts using techniques like `../../`.
`RestrictedExtensions_URIPATH`	Inspects requests whose URI path includes system file extensions that the clients shouldn't read or run. Example patterns include extensions like `.log` and `.ini`.
`RestrictedExtensions_QUERYARGUMENTS`	Inspects requests whose query arguments are system file extensions that the clients shouldn't read or run. Example patterns include extensions like `.log` and `.ini`.
`GenericRFI_QUERYARGUMENTS`	Inspects the values of all query parameters and blocks requests attempting to exploit RFI (Remote File Inclusion) in web applications. Examples include patterns like `://`.
`GenericRFI_BODY`	Inspects the values of the request body and blocks requests attempting to exploit RFI (Remote File Inclusion) in web applications. Examples include patterns like `://`.

Rule name	Description
`GenericRFI_URIPATH`	Inspects the values of the URI path and blocks requests attempting to exploit RFI (Remote File Inclusion) in web applications. Examples include patterns like `://`.
`CrossSiteScripting_COOKIE`	Inspects the value of cookie headers and blocks common cross-site scripting (XSS) patterns using the built-in XSS detection rule in AWS WAF. Example patterns include scripts like `<script>alert("hello")</script>`.
`CrossSiteScripting_QUERYARGUMENTS`	Inspects the value of query arguments and blocks common cross-site scripting (XSS) patterns using the built-in XSS detection rule in AWS WAF. Example patterns include scripts like `<script>alert("hello")</script>`.
`CrossSiteScripting_BODY`	Inspects the value of the request body and blocks common cross-site scripting (XSS) patterns using the built-in XSS detection rule in AWS WAF. Example patterns include scripts like `<script>alert("hello")</script>`.
`CrossSiteScripting_URIPATH`	Inspects the value of the URI path and blocks common cross-site scripting (XSS) patterns using the built-in XSS detection rule in AWS WAF. Example patterns include scripts like `<script>alert("hello")</script>`.

Admin protection

VendorName: `AWS`, Name: `AWSManagedRulesAdminProtectionRuleSet`, WCU: 100

The Admin protection rule group contains rules that allow you to block external access to exposed administrative pages. This might be useful if you run third-party software or want to reduce the risk of a malicious actor gaining administrative access to your application.

Rule name	Description
`AdminProtection_URIPATH`	Inspects requests for URI paths that are generally reserved for administration of a webserver or application. Example patterns include `sqlmanager`.

Known bad inputs

VendorName: `AWS`, Name: `AWSManagedRulesKnownBadInputsRuleSet`, WCU: 200

The Known bad inputs rule group contains rules to block request patterns that are known to be invalid and are associated with exploitation or discovery of vulnerabilities. This can help reduce the risk of a malicious actor discovering a vulnerable application.

Rule name	Description
`Host_localhost_HEADER`	Inspects the host header in the request for patterns indicating localhost. Example patterns include `localhost`.
`PROPFIND_METHOD`	Inspects the HTTP method in the request for `PROPFIND`, which is a method similar to `HEAD`, but with the extra intention to exfiltrate XML objects.
`ExploitablePaths_URIPATH`	Inspects the URI path for attempts to access exploitable web application paths. Example patterns include paths like `web-inf`.
`BadAuthToken_COOKIE_AUTHORIZATION`	Inspects the request for the presence of an unsolicited JSON Web Token (JWT). This protects against a malicious actor attempting to guess the secret key.

Use-case specific rule groups

Use-case specific rule groups provide incremental protection for many diverse AWS WAF use cases. Choose the rule groups that apply to your application.

SQL database

VendorName: `AWS`, Name: `AWSManagedRulesSQLiRuleSet`, WCU: 200

The SQL database rule group contains rules to block request patterns associated with exploitation of SQL databases, like SQL injection attacks. This can help prevent remote injection of unauthorized queries. Evaluate this rule group for use if your application interfaces with an SQL database.

Rule name	Description
`SQLiExtendedPatterns_QUERYARGUMENTS`	Inspects the values of all query parameters for patterns that match malicious SQL code. The patterns this rule inspects for aren't covered by the built-in AWS WAF SQL injection match statement.
`SQLi_QUERYARGUMENTS`	Uses the built-in AWS WAF SQL injection match statement to inspect the values of all query parameters for patterns that match malicious SQL code.
`SQLi_BODY`	Uses the built-in AWS WAF SQL injection match statement to inspect the request body for patterns that match malicious SQL code.
`SQLi_COOKIE`	Uses the built-in AWS WAF SQL injection match statement to inspect the request cookie header for patterns that match malicious SQL code.
`SQLi_URIPATH`	Uses the built-in AWS WAF injection match statement to inspect the request URI path for patterns that match malicious SQL code.

Linux operating system

VendorName: `AWS`, Name: `AWSManagedRulesLinuxRuleSet`, WCU: 200

The Linux operating system rule group contains rules that block request patterns associated with the exploitation of vulnerabilities specific to Linux, including Linux-specific Local File Inclusion (LFI) attacks. This can help prevent attacks that expose file contents or run code for which the attacker should not have had access. You should evaluate this rule group if any part of your application runs on Linux. You should use this rule group in conjunction with the POSIX operating system (p. 42) rule group.

Rule name	Description
`LFI_URIPATH`	Inspects the request path for attempts to exploit Local File Inclusion (LFI) vulnerabilities in web applications. Example patterns include files like `/proc/version`, which could provide operating system information to attackers.
`LFI_QUERYARGUMENTS`	Inspects the values of all query parameters for attempts to exploit Local File Inclusion (LFI) vulnerabilities in web applications. Example patterns include files like `/proc/version`, which could provide operating system information to attackers.
`LFI_BODY`	Inspects the request body for attempts to exploit Local File Inclusion (LFI) vulnerabilities in web applications. Example patterns include files like `/proc/version`, which could provide operating system information to attackers.

POSIX operating system

VendorName: `AWS`, Name: `AWSManagedRulesUnixRuleSet`, WCU: 100

The POSIX operating system rule group contains rules that block request patterns associated with the exploitation of vulnerabilities specific to POSIX and POSIX-like operating systems, including Local File Inclusion (LFI) attacks. This can help prevent attacks that expose file contents or run code for which the attacker should not have had access. You should evaluate this rule group if any part of your application runs on a POSIX or POSIX-like operating system, including Linux, AIX, HP-UX, macOS, Solaris, FreeBSD, and OpenBSD.

Rule name	Description
`UNIXShellCommandsVariables_QUERYARGUMENTS`	Inspects the values of all query parameters for attempts to exploit command injection, LFI, and path traversal vulnerabilities in web applications that run on Unix systems. Examples include patterns like `echo $HOME` and `echo $PATH`.
`UNIXShellCommandsVariables_BODY`	Inspects the request body for attempts to exploit command injection, LFI, and path traversal vulnerabilities in web applications that run on Unix systems. Examples include patterns like `echo $HOME` and `echo $PATH`.

Windows operating system

VendorName: `AWS`, Name: `AWSManagedRulesWindowsRuleSet`, WCU: 200

The Windows operating system rule group contains rules that block request patterns associated with the exploitation of vulnerabilities specific to Windows, like remote execution of PowerShell commands. This can help prevent exploitation of vulnerabilities that allow an attacker to run unauthorized commands or run malicious code. Evaluate this rule group if any part of your application runs on a Windows operating system.

Rule name	Description
`PowerShellCommands_Set1_QUERYARGUMENTS`	Inspects the values of all query parameters and blocks PowerShell command injection attempts in web applications. This inspection requires two rules, to accommodate the size of the pattern matching set. The match patterns represent PowerShell commands, for example, `Invoke-Expression`.
`PowerShellCommands_Set2_QUERYARGUMENTS`	Inspects the values of all query parameters and blocks PowerShell command injection attempts in web applications. This inspection requires two rules, to accommodate the size of the pattern matching set. The match patterns represent PowerShell commands, for example, `Invoke-Expression`.
`PowerShellCommands_Set1_BODY`	Inspects the values of the request body and blocks PowerShell command injection attempts in web applications. This inspection requires two rules, to accommodate the size of the pattern matching set. The match patterns represent PowerShell commands, for example, `Invoke-Expression`.
`PowerShellCommands_Set2_BODY`	Inspects the values of the request body and blocks PowerShell command injection attempts in web applications. This inspection requires two rules, to accommodate the size of the pattern matching set. The match patterns represent PowerShell commands, for example, `Invoke-Expression`.

PHP application

VendorName: `AWS`, Name: `AWSManagedRulesPHPRuleSet`, WCU: 100

The PHP application rule group contains rules that block request patterns associated with the exploitation of vulnerabilities specific to the use of the PHP programming language, including injection of unsafe PHP functions. This can help prevent exploitation of vulnerabilities that allow an attacker to remotely run code or commands for which they are not authorized. Evaluate this rule group if PHP is installed on any server with which your application interfaces.

Rule name	Description
`PHPHighRiskMethodsVariables_QUERYARGUMENTS`	Inspects the values of all query parameters for PHP script code injection attempts. Example

Rule name	Description
	patterns include functions like `fsockopen` and the `$_GET` superglobal variable.
`PHPHighRiskMethodsVariables_BODY`	Inspects the values of the request body for PHP script code injection attempts. Example patterns include functions like `fsockopen` and the `$_GET` superglobal variable.

WordPress application

VendorName: `AWS`, Name: `AWSManagedRulesWordPressRuleSet`, WCU: 100

The WordPress application rule group contains rules that block request patterns associated with the exploitation of vulnerabilities specific to WordPress sites. You should evaluate this rule group if you are running WordPress. This rule group should be used in conjunction with the SQL database (p. 41) and PHP application (p. 43) rule groups.

Rule name	Description
`WordPressExploitableCommands_QUERYSTRING`	Inspects the request query string for high risk WordPress commands that can be exploited in vulnerable installations or plugins. Examples patterns include commands like `do-reset-wordpress`.
`WordPressExploitablePaths_URIPATH`	Inspects the request URI path for WordPress files like `xmlrpc.php`, which are known to have easily exploitable vulnerabilities.

IP reputation rule groups

IP reputation rule groups allow you to block requests based on their source. Choose one or more of these rule groups if you want to reduce your exposure to bot traffic or exploitation attempts, or if you are enforcing geographic restrictions on your content. For bot management, see also AWS WAF Bot Control rule group (p. 45).

Amazon IP reputation list

VendorName: `AWS`, Name: `AWSManagedRulesAmazonIpReputationList`, WCU: 25

The Amazon IP reputation list rule group contains rules that are based on Amazon internal threat intelligence. This is useful if you would like to block IP addresses typically associated with bots or other threats. Blocking these IP addresses can help mitigate bots and reduce the risk of a malicious actor discovering a vulnerable application.

The rules in this rule group add labels to matching requests. For information about labeling, see AWS WAF labels on web requests (p. 54).

Rule name	Description
`AWSManagedIPReputationList`	Inspects for a list of IP addresses that have been identified as bots by Amazon threat intelligence. Label: `awswaf:managed:aws:amazon-ip-list:AWSManagedIPReputationList`.

Anonymous IP list

VendorName: `AWS`, Name: `AWSManagedRulesAnonymousIpList`, WCU: 50

The Anonymous IP list rule group contains rules to block requests from services that allow the obfuscation of viewer identity. These include requests from VPNs, proxies, Tor nodes, and hosting providers (including AWS). This rule group is useful if you want to filter out viewers that might be trying to hide their identity from your application. Blocking the IP addresses of these services can help mitigate bots and evasion of geographic restrictions.

The rules in this rule group add labels to matching requests. For information about labeling, see AWS WAF labels on web requests (p. 54).

Rule name	Description
`AnonymousIPList`	Inspects for a list of IP addresses of sources known to anonymize client information, like TOR nodes, temporary proxies, and other masking services. Label: `awswaf:managed:aws:anonymous-ip-list:AnonymousIPList`.
`HostingProviderIPList`	Inspects for a list of IP addresses from hosting and cloud providers, which are less likely to source end-user traffic. Examples include cloud providers like AWS. Label: `awswaf:managed:aws:anonymous-ip-list:HostingProviderIPList`.

AWS WAF Bot Control rule group

The Bot Control managed rule group available from AWS Managed Rules.

AWS WAF Bot Control

VendorName: `AWS`, Name: `AWSManagedRulesBotControlRuleSet`, WCU: 50

The Bot Control managed rule group contains rules to block and manage requests from bots. You are charged additional fees when you use this rule group. For more information, see AWS WAF Pricing.

In order to keep your costs down and to be sure you're managing your bot traffic as you want, use this rule group in accordance with the guidance at AWS WAF Bot Control (p. 94).

The Bot Control managed rule group generates labels for each rule. You can retrieve the labels through the API by calling `DescribeManagedRuleGroup`. The labels are listed in the `AvailableLabels` property in the response.

The Bot Control labels are generated in the following namespaces, depending on the rule:

- `bot:category:` – The category of bot, as defined by AWS WAF, for example `bot:category:search_engine:` and `bot:category:content_fetcher:`.
- `bot:name:` – The bot name, if one is available, for example `bot:name:slurp`, `bot:name:googlebot`, and `bot:name:pocket_parser`.
- `bot:` – Used to indicate a verified bot with the label `bot:verified`. This is used for common desirable bots, like `googlebot` and `bingbot`.

Bot Control uses the IP addresses from AWS WAF to verify bots. If you have verified bots that route through a proxy or load balancer, you might need to explicitly allow them. For information, see Forwarded IP address (p. 82).

- `signal:` – Used for attributes of the request that are indicative of bots that are not more commonly used or verified.

The Bot Control managed rule group applies labels to a set of verifiable bots that are commonly allowed. The rule group doesn't block this category of bots and doesn't apply any `signal:` labels. If you want, you can block them, or a subset of them, by writing a custom rule that uses the labels applied by the Bot Control managed rule group. For more information about this and examples, see AWS WAF Bot Control (p. 94).

Rule name	Description
`CategoryAdvertising`	Inspects for bots that are used for advertising purposes.
`CategoryArchiver`	Inspects for bots that are used for archiving purposes.
`CategoryContentFetcher`	Inspects for bots that are fetching content on behalf of an end user.
`CategoryHttpLibrary`	Inspects for HTTP libraries that are often used by bots.
`CategoryLinkChecker`	Inspects for bots that check for broken links.
`CategoryMiscellaneous`	Inspects for miscellaneous bots.
`CategoryMonitoring`	Inspects for bots that are used for monitoring purposes.
`CategoryScrapingFramework`	Inspects for web scraping frameworks.
`CategorySecurity`	Inspects for security-related bots.
`CategorySeo`	Inspects for bots that are used for search engine optimization.
`CategorySocialMedia`	Inspects for bots that are used by social media platforms to provide content summaries. Verified social media bots are not blocked.
`CategorySearchEngine`	Inspects for search engine bots. Verified search engines are not blocked.
`SignalAutomatedBrowser`	Inspects for indications of an automated web browser.
`SignalKnownBotDataCenter`	Inspects for data centers that are typically used by bots.
`SignalNonBrowserUserAgent`	Inspects for user agent strings that don't seem to be from a web browser.

AWS Managed Rules disclaimer

AWS Managed Rules are designed to protect you from common web threats. When used in accordance with the documentation, AWS Managed Rules rule groups add another layer of security for your applications. However, AWS Managed Rules rule groups aren't intended as a replacement for your security responsibilities, which are determined by the AWS resources that you select. Refer to the Shared Responsibility Model to ensure that your resources in AWS are properly protected.

AWS Managed Rules changelog

This section lists changes to the AWS Managed Rules for AWS WAF since their release in November, 2019.

Note
This changelog reports changes to the rules and rule groups in AWS Managed Rules for AWS WAF. It doesn't report changes to the IP address lists that are used by the rules in the IP reputation rule groups (p. 44), due to the dynamic nature of those lists.

Rule group	Rules affected	Description	Date
Amazon IP reputation list	`AWSManagedIPReputationList_ack`	Restructured the IP reputation list, removed suffixes from rule name, and added support for AWS WAF labels.	2021-05-04

Anonymous IP list	`AnonymousIPList`
	
	`HostingProviderList`
		Added support for AWS WAF labels.	2021-05-04

AWS WAF Bot Control	All	Added the Bot Control rule set.	2021-04-01
Core rule set (CRS)	`GenericRFI_QUERYARGUMENTS`	Added double URL decode.	2021-03-03
Core rule set (CRS)	`RestrictedExtensions_URIPATH`	Improved the configuration of the rules and added an extra URL decode.	2021-03-03
Admin protection	`AdminProtection_URIPATH`	Added double URL decode.	2021-03-03
Known bad inputs	`ExploitablePaths_URIPATH`	Improved the configuration of the rules and added an extra URL decode.	2021-03-03
Linux operating system	`LFI_QUERYARGUMENTS`	Improved the configuration of the rules and added an extra URL decode.	2021-03-03
Windows operating system	All	Improved the configuration of the rules.	2020-09-23

PHP application	`PHPHighRiskMethodsVariables_QUERYARGUMENTS`
	
	`PHPHighRiskMethodsVariables_BODY`
		Changed the transformation from HTML decode to URL	2020-09-16

Rule group	Rules affected	Description	Date
		decode, to improve blocking.	
POSIX operating system	`UNIXShellCommandsVariables_QUERYARGUMENTS` `UNIXShellCommandsVariables_BODY`	Changed the text transformation from HTML decode to URL decode, to improve blocking.	2020-09-16

Core rule set	`GenericLFI_QUERYARGUMENTS`
	`GenericLFI_URIPATH`
	GenericLFI_BODY
		Changed the text transformation from HTML decode to URL decode, to improve blocking.	2020-08-07

Linux operating system	`LFI_URIPATH`
	`LFI_QUERYARGUMENTS`
	`LFI_BODY`
		Changed the text transformation from HTML entity decode to URL decode, to improve detection and blocking.	2020-05-19

Anonymous IP List	All	New rule group in IP reputation rule groups (p. 44) to block requests from services that allow the obfuscation of viewer identity, to help mitigate bots and evasion of geographic restrictions.	2020-03-06
Wordpress application	`WordPressExploitableCommands_QUERYSTRING`	New rule with checks for exploitable commands in the query string.	2020-03-03

Core Rule Set (CRS)	`SizeRestrictions_QUERYSTRING`
	`SizeRestrictions_Cookie_HEADER`
	`SizeRestrictions_BODY`
	`SizeRestrictions_URIPATH`
		Adjusted the size value constraints for improved accuracy.	2020-03-03

SQL database	`SQLi_URIPATH`	The rules now check the message URI.	2020-01-23

SQL database	`SQLi_BODY`
	`SQLi_QUERYARGUMENTS`
	`SQLi_COOKIE`
		Updated text transformations.	2019-12-20

Rule group	Rules affected	Description	Date

Core Rule Set (CRS)	`CrossSiteScripting_URIPATH`
	`CrossSiteScripting_BODY`
	`CrossSiteScripting_QUERYARGUMENTS`
	`CrossSiteScripting_COOKIE`
		Updated text transformations.	2019-12-20

AWS Marketplace managed rule groups

AWS Marketplace managed rule groups are available by subscription through the AWS Marketplace console at AWS Marketplace. After you subscribe to a AWS Marketplace managed rule group, you can use it in AWS WAF. To use an AWS Marketplace rule group in an AWS Firewall Manager AWS WAF policy, each account in your organization must subscribe to it.

AWS Marketplace Rule Group Pricing

AWS Marketplace rule groups are available with no long-term contracts, and no minimum commitments. When you subscribe to a rule group, you are charged a monthly fee (prorated hourly) and ongoing request fees based on volume. For more information, see AWS WAF Pricing and the description for each AWS Marketplace rule group at AWS Marketplace.

Subscribing to AWS Marketplace managed rule groups

You can subscribe to and unsubscribe from AWS Marketplace rule groups on the AWS WAF console. If you need to, you can exclude specific rules in a managed rule group when you add it to a web ACL.

> **Important**
> To use an AWS Marketplace rule group in an AWS Firewall Manager policy, each account in your organization must first subscribe to that rule group.

To subscribe to an AWS Marketplace managed rule group

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **AWS Marketplace**.
3. In the **Available marketplace products** section, choose the name of a rule group to view the details and pricing information.
4. If you want to subscribe to the rule group, choose **Continue**.

   > **Note**
   > If you don't want to subscribe to this rule group, simply close this page in your browser.
5. Choose **Set up your account**.
6. Add the rule group to a web ACL, similar to how you add an individual rule. For more information, see Creating a web ACL (p. 22) or Editing a web ACL (p. 24).

   > **Note**
   > When adding a rule group to a web ACL, you can override the actions of rules in the rule group and of the rule group result. For more information, see Overriding the actions of a rule group or its rules (p. 20).

After you're subscribed to an AWS Marketplace rule group, you use it in your web ACLs as you do other managed rule groups. For information, see Creating a web ACL (p. 22).

Unsubscribing from AWS Marketplace managed rule groups

You can unsubscribe from AWS Marketplace rule groups on the AWS WAF console.

> **Important**
> To stop the subscription charges for an AWS Marketplace managed rule group, you must remove it from all web ACLs in AWS WAF and in any Firewall Manager AWS WAF policies, in addition to unsubscribing from it. If you unsubscribe from an AWS Marketplace managed rule group but don't remove it from your web ACLs, you will continue to be charged for the subscription.

To unsubscribe from an AWS Marketplace managed rule group

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. Remove the rule group from all web ACLs. For more information, see Editing a web ACL (p. 24).
3. In the navigation pane, choose **AWS Marketplace**.
4. Choose **Manage your subscriptions**.
5. Choose **Cancel subscription** next to the name of the rule group that you want to unsubscribe from.
6. Choose **Yes, cancel subscription**.

Troubleshooting AWS Marketplace rule groups

If you find that an AWS Marketplace rule group is blocking legitimate traffic, you can troubleshoot the problem by performing the following steps.

To troubleshoot an AWS Marketplace rule group

1. Exclude the specific rules that are blocking legitimate traffic. You can identify which rules are blocking which requests using either the AWS WAF sampled requests or AWS WAF logs. You can identify the rules by looking at the `ruleGroupId` field in the log or the `RuleWithinRuleGroup` in the sampled request. You can identify the rule in the pattern `<Seller Name>#<RuleGroup Name>#<Rule Name>`.
2. If excluding specific rules does not solve the problem, you can change the action for the AWS Marketplace rule group from **No override** to **Override to count**. This allows the web request to pass through, regardless of the individual rule actions within the rule group. This also provides you with Amazon CloudWatch metrics for the rule group.
3. After setting the AWS Marketplace rule group action to **Override to count**, contact the rule group provider's customer support team to further troubleshoot the issue. For contact information, see the rule group listing on the product listing pages on AWS Marketplace.

Contacting AWS support

For problems with AWS WAF or a rule group that is managed by AWS, contact AWS Support. For problems with a rule group that is managed by an AWS AWS Marketplace seller, contact the provider's customer support team. To find contact information, see the providers's listing on AWS Marketplace.

Managing your own rule groups

You can create your own rule group to reuse collections of rules that you either don't find in the managed rule group offerings or that you prefer to handle on your own. When you create your own rule group, you must set an immutable maximum capacity for it.

Rule groups that you create hold rules just like a web ACL does, and you add rules to a rule group in the same way as you do to a web ACL.

Topics

Creating a rule group

To create a rule group

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **Rule groups**, and then **Create rule group**.
3. Enter a name and description for the rule group. You'll use these to identify the set to manage it and use it.

   Note
   You can't change the name after you create the rule group.
4. For **Region**, choose the Region where you want to store the rule group. To use a rule group in web ACLs that protect Amazon CloudFront distributions, you must use the global setting. You can use the global setting for regional applications, too.
5. Choose **Next**.
6. Add rules to the rule group using the **Rule builder** wizard, the same as you do in web ACL management. The only difference is that you can't add a rule group to another rule group.
7. For **Capacity**, set the maximum for the rule group's use of web ACL capacity units (WCUs). This is an immutable setting. For information about WCUs, see AWS WAF Web ACL capacity units (WCU) (p. 7).

   As you add rules to the rule group, the **Add rules and set capacity** pane displays the minimum required capacity, which is based on the rules that you've already added. You can use this and your future plans for the rule group to help estimate the capacity that the rule group will require.
8. Review the settings for the rule group, and choose **Create**.

Using your rule group in a Web ACL

To use a rule group in a web ACL, on the console, when you add or update the rules in your web ACL, on the **Add rules and rule groups** page, choose **Add rules**, and then choose **Add my own rules and rule groups**. Then choose **Rule group** and select your rule group from the list.

In your web ACL, you can alter the behavior of a rule group and its rules by setting the individual rule actions to count and by overriding the resulting rule group action to count. This can help you do things like test a rule group, identify false positives from rules in a rule group, and customize how a managed rule group handles your requests. For more information about these options, see Overriding the actions of a rule group or its rules (p. 20).

If your rule group contains a rate-based statement, each web ACL where you use the rule group has its own separate rate tracking and management for the rate-based rule, independent of any other web ACL where you use the rule group. For more information, see Rate-based rule statement (p. 68).

Eventual consistency

When you make changes to web ACLs or web ACL components, like rules and rule groups, AWS WAF propagates the changes everywhere that the web ACL and its components are stored and used. Your changes are applied within seconds, but there might be a brief period of inconsistency when the changes have arrived in some places and not in others. So, for example, if you add an IP address to an IP set that's

referenced by a blocking rule in a web ACL, the new address might briefly be blocked in one area while still allowed in another. This temporary inconsistency can occur when you first associate a web ACL with an AWS resource and when you change a web ACL that is already associated with a resource. Generally, any inconsistencies of this type last only a few seconds.

Deleting a rule group

Follow the guidance in this section to delete a rule group.

Deleting referenced sets and rule groups

When you delete an entity that you can use in a web ACL, like an IP set, regex pattern set, or rule group, AWS WAF checks to see if the entity is currently being used in a web ACL. If it finds that it is in use, AWS WAF warns you. AWS WAF is almost always able to determine if an entity is being referenced by a web ACL. However, in rare cases it might not be able to do so. If you need to be sure that nothing is currently using the entity, check for it in your web ACLs before deleting it. If the entity is a referenced set, also check that no rule groups are using it.

To delete a rule group

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **Rule groups**.
3. Choose the rule group that you want to delete, and then choose **Delete**.

AWS WAF rules

In every rule group and every web ACL, rules define how to inspect web requests and what to do when a web request matches the inspection criteria. Each rule requires one top-level statement, which might contain nested statements at any depth, depending on the rule and statement type.

The inspection instructions are included in the JSON format as rule statements and the action in rule actions.

You use the rules in a web ACL to define how to inspect and handle HTTP(S) web requests based on criteria like the following:

- Scripts that are likely to be malicious. Attackers embed scripts that can exploit vulnerabilities in web applications. This is known as cross-site scripting (XSS).
- IP addresses or address ranges that requests originate from.
- Country or geographical location that requests originate from.
- Length of specified part 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. You can also use regular expressions (regex) to specify these strings.
- Labels that prior rules in the web ACL have added to the request.

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

In addition to statements like the ones in the preceding list, which provide web request inspection criteria, AWS WAF supports logical statements for `AND`, `OR`, and `NOT` that you use to combine statements in a rule.

For example, based on recent requests that you've seen from an attacker, you might create a rule with a logical `AND` statement that combines the following nested statements:

- 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.

In this case, all the statements need to result in a match for the top-level `AND` statement to match.

Rules don't exist in AWS WAF on their own. They aren't AWS resources, and they don't have Amazon Resource Names (ARNs). You can access a rule by name in the rule group or web ACL where it's defined. You can manage rules and copy them to other web ACLs by using the JSON format of the rule group or web ACL that contains the rule. Or you can manage them through the AWS WAF console **Rule Builder**, which is available for web ACLs and rule groups.

Topics

AWS WAF rule name

You must assign a unique name to every rule in your web ACL or rule group.

The name can contain only the characters `A-Z`, `a-z`, `0-9`, - (hyphen), and _ (underscore). You can't change the name of a rule after you create it in your rule group or web ACL.

AWS WAF rule action

The rule action tells AWS WAF what to do with a web request when it matches the criteria defined in the rule. You can optionally add custom behavior to each rule action.

Here are the rule action options:

- **Count** – AWS WAF counts the request but doesn't determine whether to allow it or block it. With this action, AWS WAF continues processing the remaining rules in the web ACL. You can insert custom headers into the request and you can add labels that other rules can match against.
- **Allow** – AWS WAF allows the request to be forwarded to the protected AWS resource for processing and response. You can insert custom headers into the request before forwarding it to the protected resource.
- **Block** – AWS WAF blocks the request. By default, the AWS resource responds with an HTTP `403 (Forbidden)` status code, but you can customize the response. When AWS WAF blocks a request, the block action settings determine the response that the protected resource sends back to the client.

For information about customizing requests and responses, see Customizing web requests and responses in AWS WAF (p. 89).

For information about adding labels to matching requests, see AWS WAF labels on web requests (p. 54).

You can override rule actions when you add them to a web ACL. When you do this, the rule runs with the action set to count. For more information about how web ACL and rule settings interact, see Web ACL rule and rule group evaluation (p. 19).

AWS WAF labels on web requests

A label is metadata that a rule can add to matching web requests. Rules can also match against labels when they inspect web requests. Labels allow a matching rule to communicate results to the rules that are evaluated later in the same web ACL. You can add labels from any rule except for rule group reference statements.

When a web request matches a rule, AWS WAF adds the rule's labels to the request. The labels remain available on the request as long as AWS WAF is evaluating it against the web ACL. Rules that run later in the web ACL can match against the label by using a label match statement. For more information about the label match statement, see Label match rule statement (p. 66).

Common use cases for AWS WAF labels include the following:

- **Evaluating a web request against multiple rule statements before taking action on the request** – After a match is found with a rule in a web ACL, AWS WAF continues evaluating against the web ACL only if the matching rule action is count. Labels allow you to evaluate and collect information for multiple rules before taking an action of allow or block on the web request. To do this, you change the actions for your existing rules to count and add labels to them. Use the labels to indicate the match and the action that you want to take on the request. The rules that you modify in this way can all run and provide information about the matches that they find, to destinations like logs and metrics. Then, in a final additional rule, you can evaluate the labels that were applied and determine how to handle the request.
- **Reusing logic across multiple rules** – If you need to reuse the same logic across multiple rules, you can use labels to single-source the logic and just test for the results. When you have multiple complex rules that use a common subset of nested rule statements, duplicating the common rule set across your complex rules can be time consuming and error prone. With labels, you can create a new rule with the common rule subset that counts matching requests and adds a label to them. You add the new rule to your web ACL so that it runs before your original complex rules. Then, in your original rules, you replace the shared rule subset with a single rule that checks for the label.

  For example, say you have multiple rules that you want to only apply to your login paths. Rather than have each rule specify the same logic for matching potential login paths, you can implement one rule that contains that logic and have the rule add a label indicating that the request is on a login path. In your web ACL, give this new rule a lower numeric priority setting than your original rules. Then, in your original rules, replace the shared logic with a check for the presence of the label.
- **Creating exceptions to rules in rule groups** – This option is particularly useful for managed rule groups, which you can't view or alter. For some managed rule groups, the rules add labels to matching web requests to indicate the rules that matched and, possibly, to provide additional information about the match. When you use a rule group that adds labels to requests in this way, you can customize the actions of the rule group by placing all of the rules in count mode, and create custom rules to run after the rule group. These custom rules handle the request based on the rule group's labels.

How labeling works

When a rule matches a web request, if the rule has labels defined, AWS WAF adds the labels to the request. Rules that are evaluated after the matching rule in the same web ACL have access to the labels that the rule has added, and can match against them.

- Any rule that's included in a single web ACL can access labels that have been added by any rule that has already run in the same web ACL. This includes rules that are defined directly inside the web ACL and those inside rule groups that are used in the web ACL. Labels don't persist with the web request after the web ACL evaluation ends.
- In order for other rules to match against a label that your rule adds, your rule action must be set to count. This way, the web ACL evaluation doesn't terminate, allowing subsequent rules in the web

ACL to run their label matching criteria. For more information about rule actions, see AWS WAF rule action (p. 53).

- Labels emit Amazon CloudWatch metrics. For information, see Monitoring with Amazon CloudWatch (p. 370).
- AWS WAF records labels in the logs. You can use labels, along with the rule action, to filter the logs that AWS WAF records. For information, see Logging web ACL traffic information (p. 107).

Label syntax and naming requirements

A label is a string made up of a prefix, optional namespaces, and a name. The components of a label are delimited with a colon. Labels have the following requirements and characteristics:

- Labels are case-sensitive.
- Each label namespace or label name can have up to 128 characters.
- You can specify up to five namespaces in a label.
- Components of a label are separated by colon (`:`).
- You can't use the following reserved strings in the namespaces or name that you specify for a label: `awswaf`, `aws`, `waf`, `rulegroup`, `webacl`, `regexpatternset`, `ipset`, and `managed`.

Label syntax

The label prefix varies between the labels that you define in your rules and the ones that are defined in managed rule groups. For information about these rule group types, see Rule groups (p. 30).

- **Your labels** – The following shows the full label syntax for labels that you create in your web ACL and rule group rules. The entity types are `rulegroup` and `webacl`.

```
awswaf:<entity owner account id>:<entity type>:<entity name>:<custom
 namespace>:...:<label name>
```

  - Label namespace prefix: `awswaf:<entity owner account id>:<entity type>:<entity name>:`
  - Custom namespace additions: `<custom namespace>:…:`

  When you define a label for a rule in a rule group or web ACL, you control the custom namespace strings and the label name. The rest is generated for you by AWS WAF. AWS WAF automatically prefixes all labels with `awswaf` and the account and web ACL or rule group entity settings.
- **Managed rule group labels** – The following shows the full label syntax for labels that are created by rules in managed rule groups.

```
awswaf:managed:<vendor>:<rule group name>:<custom namespace>:...:<label name>
```

  - Label namespace prefix: `awswaf:managed:<vendor>:<rule group name>:`
  - Custom namespace additions: `<custom namespace>:…:`

  Some managed rule groups add labels to requests, but not all. The AWS WAF Bot Control managed rule group adds labels. For information about Bot Control, see AWS WAF Bot Control (p. 94). For general information about managed rule groups, see Managed rule groups (p. 30).

Label examples

The following example labels are defined by rules in a rule group named `testRules` that belongs to the account, 111122223333.

```
awswaf:111122223333:rulegroup:testRules:testNS1:testNS2:LabelNameA
```

```
awswaf:111122223333:rulegroup:testRules:testNS1:LabelNameQ
```

```
awswaf:111122223333:rulegroup:testRules:LabelNameZ
```

The following listing shows an example label specification in JSON. These label names include custom namespace strings before the ending label name.

```
Rule: {
    Name: "label_rule",
    Statement: {...}
    RuleLabels: [
        Name: "header:encoding:utf8",
        Name: "header:user_agent:firefox"
    ],
    Action: { Count: {} }
}
```

Note
You can access this type of listing in the console through the rule JSON editor.

If you run the rule above in the same rule group and account as the preceding label examples, the resulting, fully qualified labels would be the following:

```
awswaf:111122223333:rulegroup:testRules:header:encoding:utf8
```

```
awswaf:111122223333:rulegroup:testRules:header:user_agent:firefox
```

Adding a label to matching web requests

When you define a label for a rule, AWS WAF adds the label to requests that match the rule. You define a label in a rule by specifying the custom namespace strings and name to append to the label namespace prefix. AWS WAF derives the prefix from the context in which you define the rule. For information about this, see the label syntax information under Label syntax and naming requirements (p. 55).

Except for the following, you can add labels to any rule and AWS WAF will add your labels to any web request that matches the rule match statement:

- For rate-based rules, labels are only added to web requests for a particular IP address while that IP address is blocked by AWS WAF due to rate limiting. For information about rate-based rules, see Rate-based rule statement (p. 68).
- Labels aren't allowed in statements that reference rule groups. If you try to add a label to a rule group rule statement through the API, the operation throws a validation exception. For information about these statement types, see Managed rule group statement (p. 66) and Rule group statement (p. 71).

WCUs – 1 WCU for every 5 labels that you define in your web ACL or rule group rules.

Where to find this

- **Rule builder** on the console – Under the rule's **Action** settings, under **Label**.

- **API data type** – `Rule RuleLabels`

Matching against a label

You can use a label match statement to evaluate web request labels. You can match against *Label*, which requires the label name, or against *Namespace*, which requires a namespace specification. For either label or namespace, you can optionally include preceding namespaces and the prefix in your specification. For general information about this statement type, see Label match rule statement (p. 66).

A label's prefix defines the context of the rule group or web ACL where the label's rule is defined. In a rule's label match statement, if your label or namespace match string doesn't specify the prefix, AWS WAF uses the prefix for the label match rule.

- Labels for rules that are defined directly inside a web ACL have a prefix that specifies the web ACL context.
- Labels for rules that are inside a rule group have a prefix that specifies the rule group context. This could be your own rule group or a rule group that's managed for you.

For information about this, see label syntax under Label syntax and naming requirements (p. 55).

If you want to match against a rule that's in a different context than the context of your rule, you must provide the prefix in your match string. For example, if you want to match against labels that are added by rules in a managed rule group, you could add a rule in your web ACL with a label match statement whose match string specifies the rule group's prefix followed by your additional match criteria.

In the match string for the label match statement, you specify either a label or a namespace:

- **Label** – The label specification for a match consists of the ending part of the label. You can include any number of the contiguous namespaces that immediately precede the label name followed by the name. You can also provide the fully qualified label by starting the specification with the prefix.

  Example specifications:

  - `testNS1:testNS2:LabelNameA`
  - `awswaf:managed:aws:managed-rule-set:testNS1:testNS2:LabelNameA`

- **Namespace** – The namespace specification for a match consists of any contiguous subset of the label specification excluding the name. You can include the prefix and you can include one or more namespace strings.

  Example specifications:

  - `testNS1:testNS2:`
  - `awswaf:managed:aws:managed-rule-set:testNS1:`

Label match examples

This section provides examples of match specifications, for the label match rule statement.

> **Note**
> These JSON listings were created in the console by adding a rule to a web ACL with the label match specifications and then editing the rule and switching to the **Rule JSON editor**. You can also get the JSON for a rule group or web ACL through the APIs or the command line interface.

Topics

Match against a local label

The following JSON listing shows a label match statement for a label that's been added to the web request locally, in the same context as this rule.

```
Rule: {
    Name: "match_rule",
    Statement: {
        LabelMatchStatement: {
            Scope: "LABEL",
            Key: "header:encoding:utf8"
        }
    },
    RuleLabels: [
        ...generate_more_labels...
    ],
    Action: { Block: {} }
}
```

If you use this match statement in account 111122223333, in a rule that you define for web ACL `testWebACL`, it would match the following labels.

```
awswaf:111122223333:webacl:testWebACL:header:encoding:utf8
```

```
awswaf:111122223333:webacl:testWebACL:testNS1:testNS2:header:encoding:utf8
```

It wouldn't match the following label, because the label string isn't an exact match.

```
awswaf:111122223333:webacl:testWebACL:header:encoding2:utf8
```

It wouldn't match the following label, because the context isn't the same, so the prefix doesn't match. This is true even if you added the rule group `productionRules` to the web ACL `testWebACL`, where the rule is defined.

```
awswaf:111122223333:rulegroup:productionRules:header:encoding:utf8
```

Match against a label from another context

The following JSON listing shows a label match rule that matches against a label from a rule inside a user-created rule group. The prefix is required in the specification for all rules running in the web ACL that aren't part of the named rule group. This example label specification matches only the exact label.

```
Rule: {
    Name: "match_rule",
    Statement: {
        LabelMatchStatement: {
            Scope: "LABEL",
            Key: "awswaf:111122223333:rulegroup:testRules:header:encoding:utf8"
        }
    },
```

```
    RuleLabels: [
        ...generate_more_labels...
    ],
    Action: { Block: {} }
}
```

Match against a managed rule group label

This is a special case of matching against a label that's from another context than that of the match rule. The following JSON listing shows a label match statement for a managed rule group label. This matches only the exact label that's specified in the label match statement's key setting.

```
Rule: {
    Name: "match_rule",
    Statement: {
        LabelMatchStatement: {
            Scope: "LABEL",
            Key: "awswaf:managed:aws:managed-rule-set:header:encoding:utf8"
        }
    },
    RuleLabels: [
        ...generate_more_labels...
    ],
    Action: { Block: {} }
}
```

Match against a local namespace

The following JSON listing shows a label match statement for a local namespace.

```
Rule: {
    Name: "match_rule",
    Statement: {
        LabelMatchStatement: {
            Scope: "NAMESPACE",
            Key: "header:encoding:"
        }
    },
    Labels: [
        ...generate_more_labels...
    ],
    Action: { Block: {} }
}
```

Similar to the local `Label` match, if you use this statement in account 111122223333, in a rule that you define for web ACL `testWebACL`, it would match the following label.

```
awswaf:111122223333:webacl:testWebACL:header:encoding:utf8
```

It wouldn't match the following label, because the account isn't the same, so the prefix doesn't match.

```
awswaf:444455556666:webacl:testWebACL:header:encoding:utf8
```

The prefix also doesn't match any labels applied by managed rule groups, like the following.

```
awswaf:managed:aws:managed-rule-set:header:encoding:utf8
```

Match against a managed rule group namespace

The following JSON listing shows a label match statement for a managed rule group namespace. For a rule group that you own, you'd also need to provide the prefix in order to match for a namespace that's outside of the rule's context.

```
Rule: {
    Name: "match_rule",
    Statement: {
        LabelMatchStatement: {
            Scope: "NAMESPACE",
            Key: "awswaf:managed:aws:managed-rule-set:header:"
        }
    },
    RuleLabels: [
        ...generate_more_labels...
    ],
    Action: { Block: {} }
}
```

This specification matches against the following example labels.

```
awswaf:managed:aws:managed-rule-set:header:encoding:utf8
```

```
awswaf:managed:aws:managed-rule-set:header:encoding:unicode
```

It doesn't match the following label.

```
awswaf:managed:aws:managed-rule-set:query:badstring
```

AWS WAF rule statements

Rule statements are the part of a rule that tells AWS WAF how to inspect a web request. When AWS WAF finds the inspection criteria in a web request, we say that the web request matches the statement. Every rule statement specifies what to look for and how, according to the statement type.

Every rule in AWS WAF has a single top-level rule statement, which can contain other statements. Rule statements can be very simple. For example, you could have a statement that provides a set of originating countries to check your web requests for. Rule statements can also be very complex. For example, you could have a statement that combines many other statements with logical `AND`, `OR`, and `NOT` statements.

Web ACLs can also contain rule statements that just reference rule groups. On the console, you don't see these represented as rule statements, but every web ACL has a JSON format representation. In the JSON, you can see these special types of rule statements. For rules of any complexity, managing your web ACL using the JSON editor is the easiest way to go. You can retrieve the complete configuration for a web ACL in JSON format, modify it as you need, and then provide it to AWS WAF through the console, API, or CLI. The same is true for rule groups that you manage on your own.

Nesting rule statements

AWS WAF supports nesting for many rule statements, but not for all. For example, you can't nest a rule group statement inside of another statement. You need to use nesting for some scenarios, such as scope-down statements and logical statements. The rule statement lists and rule details that follow describe the nesting capabilities and requirements for each category and rule.

Topics

Rule statements list

This section describes the statements that you can add to a rule and provides some guidelines for calculating web ACL capacity units (WCU) usage for each.

This page groups the rule statements by category, provides a high-level description for each, and provides a link to a section with more information for the statement type.

Match statements

Match statements compare the web request or its origin against conditions that you provide. For many statements of this type, AWS WAF compares a specific component of the request for matching content.

Match statements are nestable. You can nest them inside logical rule statements and use them in scope-down statements.

Match Statement	Description	WCUs
Geographic match (p. 64)	Inspects the request's country of origin.	1
IP set match (p. 65)	Inspects the request against a set of IP addresses and address ranges.	1 for most cases. If you configure the statement to use a header with forwarded IP addresses and specify a position in the header of `Any`, then the WCUs are 5.
Label match rule statement (p. 66)	Inspects the request for labels that have been added by other rules in the same web ACL.	1
Regex match rule statement (p. 69)	Compares a regex pattern against a specified request component.	3, as a base cost. If you use the request component **All query parameters**, add 10 WCUs. If you use the request component **JSON body**, double the base cost WCUs. For each **Text transformation** that you apply, add 10 WCUs.
Regex pattern set (p. 70)	Compares regex patterns against a specified request component.	25 per pattern set, as a base cost. If you use the request component **All query parameters**, add 10 WCUs. If you use the request component **JSON body**, double the base cost WCUs. For each **Text**

Match Statement	Description	WCUs
		transformation that you apply, add 10 WCUs.
Size constraint (p. 71)	Checks size constraints against a specified request component.	1, as a base cost. If you use the request component **All query parameters**, add 10 WCUs. If you use the request component **JSON body**, double the base cost WCUs. For each **Text transformation** that you apply, add 10 WCUs.
SQLi attack (p. 72)	Inspects for malicious SQL code in a specified request component.	20, as a base cost. If you use the request component **All query parameters**, add 10 WCUs. If you use the request component **JSON body**, double the base cost WCUs. For each **Text transformation** that you apply, add 10 WCUs.
String match (p. 73)	Compares a string to a specified request component.	The base cost depends on the type of string match and is between 1 and 10. If you use the request component **All query parameters**, add 10 WCUs. If you use the request component **JSON body**, double the base cost WCUs. For each **Text transformation** that you apply, add 10 WCUs.
XSS scripting attack (p. 74)	Inspects for cross-site scripting attacks in a specified request component.	40, as a base cost. If you use the request component **All query parameters**, add 10 WCUs. If you use the request component **JSON body**, double the base cost WCUs. For each **Text transformation** that you apply, add 10 WCUs.

Logical rules statements

Logical rules statements allow you to combine other statements or negate their results. Every logical rule statement takes at least one nested statement.

To logically combine or negate rule statement results, you nest the statements under logical rule statements.

Note
The visual editor on the console supports one level of rule statement nesting, which works for many needs. To nest more levels, edit the JSON representation of the rule on the console or use the APIs.

Logical rules statements are nestable. You can nest them inside other logical rule statements and use them in scope-down statements. For information about scope-down statements, see Scope-down statements (p. 81).

Logical Statement	Description	WCUs
AND logic (p. 64)	Combines nested statements with AND logic.	Based on nested statements
NOT logic (p. 67)	Negates the results of a nested statement.	Based on nested statement
OR logic (p. 67)	Combines nested statements with OR logic.	Based on nested statements

Complex statements

AWS WAF supports rate-based and rule group statements. You can't nest these statement types inside other rule statements. For some of these statements, you can narrow the scope of the requests that they inspect by adding a scope-down statement.

Statement	Description	WCUs
Managed rule group (p. 66)	Runs the rules that are defined in the specified managed rule group. You can narrow the scope of requests that you evaluate with the rule group by adding a scope-down statement. You cannot nest a managed rule group statement inside any other statement type.	Defined by the rule group, plus any additional WCUs for the scope-down statement.
Rule group (p. 71)	Runs the rules that are defined in a rule group that you manage. You cannot nest a rule group statement inside any other statement type.	You define the WCU limit for the rule group when you create it.
Rate-based (p. 68)	Tracks the rate of requests from individual IP addresses and temporarily blocks addresses while they are sending too many requests. You can narrow the scope of requests that you evaluate	2, plus any additional WCUs for the scope-down statement.

Statement	Description	WCUs
	with a rate-based statement by adding a scope-down statement. You cannot nest a rate-based statement under any rule statement. You can define a rate-based statement inside a rule group that you manage.	

`AND` rule statement

The `AND` rule statement combines nested statements with a logical `AND` operation, so all nested statements must match for the `AND` statement to match. This requires at least one nested statement.

Nestable – You can nest this statement type.

WCUs – Depends on the nested statements.

Where to find this

- **Rule builder** on the console – For **If a request**, choose **matches all the statements (AND)**, and then fill in the nested statements.
- **API statement** – `AndStatement`

Geographic match rule statement

To allow or block web requests based on country of origin, create one or more geographical, or geo, match statements.

> **Note**
> If you use the CloudFront geo restriction feature to block a country from accessing your content, any request from that country is blocked and is not forwarded to AWS WAF. So if you want to allow or block requests based on geography plus other AWS WAF criteria, you should *not* use the CloudFront geo restriction feature. Instead, you should use an AWS WAF geo match condition.

You can use this to block access to your site from specific countries or to only allow access from specific countries. If you want to allow some web requests and block others based on country of origin, add a geo match statement for the countries that you want to allow and add a second one for the countries that you want to block.

You can use geo match statements with other AWS WAF statements to build sophisticated filtering. For example, to block certain countries, but still allow requests from a specific set of IP addresses in that country, you could create a rule with the action set to `Block` and the following nested statements:

- `AND` statement
  - Geo match statement listing the countries that you want to block
  - `NOT` statement
    - IP set statement that specifies the IP addresses that you want to allow through

As another example, if you want to prioritize resources for users in a particular country, you could create a different rate-based rules statement for each geo match condition. Set a higher rate limit for users in the preferred country and set a lower rate limit for all other users.

AWS WAF determines the country of origin by resolving the IP address of the web request's origin. If you want to instead use an IP address from an alternate header, like `X-Forwarded-For`, enable forwarded IP configuration.

Nestable – You can nest this statement type.

WCUs – 1 WCU.

This statement uses the following settings:

- **Geo match** – An array of country codes to compare for a geo match. These must be two-character country codes, for example, `[ "US", "CN" ]`, from the alpha-2 country ISO codes of the ISO 3166 international standard.

  Each code must be one or two characters long.
- **(Optional) Forwarded IP configuration** – By default, AWS WAF uses the IP address in the web request origin to determine country of origin. Alternatively, you can configure the rule to use a forwarded IP in an HTTP header like `X-Forwarded-For` instead. AWS WAF uses the first IP address in the header. With this configuration, you also specify a fallback behavior to apply to a web request with a malformed IP address in the specified header. The fallback behavior sets the matching result for the request, to match or no match. For more information, see Forwarded IP address (p. 82).

Where to find this

- **Rule builder** on the console – For **Request option**, choose **Originates from a country in**.
- **API statement** – `GeoMatchStatement`

IP set match rule statement

The IP set match statement inspects the IP address of a web request against a set of IP addresses and address ranges. Use this to allow or block web requests based on the IP addresses that the requests originate from. By default, AWS WAF uses the IP address from the web request origin, but you can configure the rule to use an HTTP header like `X-Forwarded-For` instead.

AWS WAF supports all IPv4 and IPv6 CIDR ranges except for `/0`. For more information about CIDR notation, see the Wikipedia entry Classless Inter-Domain Routing. An IP set can hold up to 10,000 IP addresses or IP address ranges to check.

> **Note**
> Each IP set match rule references an IP set, which you create and maintain independent of your rules. This allows you to use the single set in multiple rules. When you update the referenced IP set, AWS WAF automatically updates all rules that reference it.
> For information about creating and managing an IP set, see Creating and managing an IP set (p. 85).

When you add or update the rules in your rule group or web ACL, choose the option **IP set** and select the name of the IP set that you want to use.

Nestable – You can nest this statement type.

WCUs – 1 WCU for most. If you configure the statement to use forwarded IP addresses and specify a position of `ANY`, the WCU usage is 5.

This statement uses the following settings:

- **IP set specification** – Choose the IP set that you want to use from the list or create a new one.

- **(Optional) Forwarded IP configuration** – An alternate forwarded IP header name to use in place of the request origin. You specify whether to match against the first, last, or any address in the header. You also specify a fallback behavior to apply to a web request with a malformed IP address in the specified header. The fallback behavior sets the matching result for the request, to match or no match. For more information, see Forwarded IP address (p. 82).

Where to find this

- **Rule builder** on the console – For **Request option**, choose **Originates from an IP address in**.
- **Add my own rules and rule groups** page on the console – Choose the **IP set** option.
- **API statement** – `IPSetReferenceStatement`

Label match rule statement

The label match statement inspects the labels that are on the web request against a string specification. The labels that are available to a rule for inspection are those that have already been added to the web request by other rules in the same web ACL evaluation. Labels don't persist outside of the web ACL evaluation.

> **Note**
> A label match statement can only see labels from rules that are evaluated earlier in the web ACL. For information about how AWS WAF evaluates the rules and rule groups in web ACL processing, see Processing order of rules and rule groups in a web ACL (p. 19).

For more information about labels, see AWS WAF labels on web requests (p. 54).

Nestable – You can nest this statement type.

WCUs – 1 WCU

This statement uses the following settings:

- **Match scope** – Set this to **Label** to match against the label name and, optionally, the preceding namespaces and prefix. Set this to **Namespace** to match against some or all of the namespace specifications and, optionally, the preceding prefix.
- **Key** – The string that you want to match against. If you specify a namespace match scope, this should only specify namespaces and optionally the prefix, with an ending colon. If you specify a label match scope, this must include the label name and can optionally include preceding namespaces and prefix.

For more information about these settings, see Matching against a label (p. 57) and Label match examples (p. 57).

Where to find this

- **Rule builder** on the console – For **Request option**, choose **Has label**.
- **API statement** – `LabelMatchStatement`

Managed rule group statement

The managed rule group rule statement adds a reference in your web ACL rules list to a managed rule group. You don't see this option under your rule statements on the console, but when you work with the JSON format of your web ACL, any managed rule groups that you've added show up under the web ACL rules as this type.

A managed rule group is either an AWS Managed Rules rule group, most of which are free for AWS WAF customers, or a AWS Marketplace managed rule group. You can subscribe to AWS Marketplace managed

rule groups and the subscription AWS Managed Rules rule groups through AWS Marketplace. For more information, see Managed rule groups (p. 30).

When you add a rule group to a web ACL, you can exclude individual rules in the group from running, and you can override the actions of all rules in the rule group, to count only. For more information, see Web ACL rule and rule group evaluation (p. 19).

You can narrow the scope of the requests that AWS WAF evaluates with the rule group. To do this, you add a scope-down statement inside the rule group statement. For information about scope-down statements, see Scope-down statements (p. 81). This can help you manage how the rule group affects your traffic and can help you contain costs associated with traffic volume when you use the rule group. For information and examples for using scope-down statements with the AWS WAF Bot Control managed rule group, see AWS WAF Bot Control (p. 94).

Not nestable – You can't nest this statement type inside other statements, and you can't include it in a rule group. You can include it directly in a web ACL.

(Optional) Scope-down statement – This rule type takes an optional scope-down statement, to narrow the scope of the requests that the rule group evaluates. For more information, see Scope-down statements (p. 81).

WCUs – Set for the rule group at creation.

Where to find this

- **Console** – During the process of creating a web ACL, on the **Add rules and rule groups** page, choose **Add managed rule groups**, and then find and select the rule group that you want to use.
- **API statement** – `ManagedRuleGroupStatement`

`NOT` rule statement

The `NOT` rule statement logically negates the results of a single nested statement, so the nested statements must not match for the `NOT` statement to match, and vice versa. This requires one nested statement.

For example, if you want to block requests that don't originate in a specific country, create a `NOT` statement with action set to block, and nest a geographic match statement that specifies the country.

Nestable – You can nest this statement type.

WCUs – Depends on the nested statement.

Where to find this

- **Rule builder** on the console – For **If a request**, choose **doesn't match the statement (NOT)**, and then fill in the nested statement.
- **API statement** – `NotStatement`

`OR` rule statement

The `OR` rule statement combines nested statements with `OR` logic, so one of the nested statements must match for the `OR` statement to match. This requires at least one nested statement.

For example, if you want to block requests that come from a specific country or that contain a specific query string, you could create an `OR` statement and nest in it a geographic match statement for the country and a string match statement for the query string.

If instead you want to block requests that *don't* come from a specific country or that contain a specific query string, you would modify the previous `OR` statement to nest the geographics match statement one level lower, inside a `NOT` statement. This level of nesting requires you to use the JSON formatting, because the console supports only one level of nesting.

Nestable – You can nest this statement type.

WCUs – Depends on the nested statements.

Where to find this

- **Rule builder** on the console – For **If a request**, choose **matches at least one of the statements (OR)**, and then fill in the nested statements.
- **API statement** – `OrStatement`

Rate-based rule statement

A rate-based rule tracks the rate of requests for each originating IP address, and triggers the rule action on IPs with rates that go over a limit. You set the limit as the number of requests per 5-minute time span. You can use this type of rule to put a temporary block on requests from an IP address that's sending excessive requests. By default, AWS WAF aggregates requests based on the IP address from the web request origin, but you can configure the rule to use an IP address from an HTTP header, like `X-Forwarded-For`, instead.

AWS WAF tracks and manages web requests separately for each instance of a rate-based rule that you use. For example, if you provide the same rate-based rule settings in two web ACLs, each of the two rule statements represents a separate instance of the rate-based rule and gets its own tracking and management by AWS WAF. If you define a rate-based rule inside a rule group, and then use that rule group in multiple places, each use creates a separate instance of the rate-based rule that gets its own tracking and management by AWS WAF.

When the rule action triggers, AWS WAF applies the action to additional requests from the IP address until the request rate falls below the limit. It can take a minute or two for the action change to go into effect.

You can narrow the scope of the requests that AWS WAF counts. To do this, you nest another, scope-down statement inside the rate-based statement. Then, AWS WAF only counts requests that match the scope-down statement. For information about scope-down statements, see Scope-down statements (p. 81).

For example, based on recent requests that you've seen from an attacker in the United States, you might create a rate-based rule with the following scope-down statement:

- AND rule statement that contains the following, second level of nested statements:
  - A geo-match match statement that specifies requests originating in the United States.
  - A string match statement that searches in the `User-Agent` header for the string `BadBot`.

Let's say that you also set a rate limit of 1,000. For each IP address, AWS WAF counts requests that meet both of the conditions. Requests that don't meet both conditions aren't counted. If the count for an IP address exceeds 1,000 requests in any 5-minute time span, the rule's action triggers against that IP address.

As another example, you might want to limit requests to the login page on your website. To do this, you could create a rate-based rule with the following nested string match statement:

- The **Request option** is `URI`.
- The **Match Type** is `Starts with`.

- A **Strings to match** is `login`.

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.

Not nestable – You can't nest this statement type inside other statements. You can include it directly in a web ACL and in a rule group.

(Optional) Scope-down statement – This rule type takes an optional scope-down statement, to narrow the scope of the requests that the rate-based statement tracks. For more information, see Scope-down statements (p. 81).

WCUs – 2 plus any additional WCUs for a nested statement.

This statement uses the following optional setting:

- **(Optional) Forwarded IP configuration** – By default, AWS WAF aggregates on the IP address in the web request origin, but you can instead configure the rule to use a forwarded IP address in an HTTP header like `X-Forwarded-For`. AWS WAF uses the first IP address in the header. With this configuration, you also specify a fallback behavior to apply to a web request with a malformed IP address in the specified header. The fallback behavior sets the matching result for the request, to match or no match. For more information, see Forwarded IP address (p. 82).

Where to find this

- **Rule builder** in your web ACL, on the console – Under **Rule**, for **Type**, choose **Rate-based rule**.
- **API statement** – `RateBasedStatement`

Regex match rule statement

A regex match statement instructs AWS WAF to match a request component against a single regular expression (regex). A web request matches the statement if the request component matches the regex that you specify.

This statement type is a good alternative to the Regex pattern set match rule statement (p. 70) for situations where you want to combine your matching criteria using mathematical logic. For example, if you want a request component to match against some regex patterns and to not match against anothers, you can combine the regex match statements using the `AND` rule statement (p. 64) and the `NOT` rule statement (p. 67).

Nestable – You can nest this statement type.

WCUs – 3 WCUs, as a base cost. If you use the request component **All query parameters**, add 10 WCUs. If you use the request component **JSON body**, double the base cost WCUs. For each **Text transformation** that you apply, add 10 WCUs.

This statement type operates on a web request component, and requires the following request component settings:

- **Request components** – The part of the web request to inspect, for example, a query string or the body.

  Note
  If you use the request component **Body** or **JSON body**, only the first 8 KB (8,192 bytes) of the request body are forwarded to AWS WAF for inspection. AWS WAF does not support inspecting the entire contents of web requests whose bodies exceed 8 KB.

  For information about web request components, see Request component (p. 75).

- **Optional text transformations** – Transformations that you want AWS WAF to perform on the request component before inspecting it. For example, you could transform to lowercase or normalize white space. If you specify more than one transformation, AWS WAF processes them in the order listed. For information, see Text transformations (p. 78).

Where to find this

- **Rule builder** on the console – For **Match type**, choose **Matches regular expression**.
- **API statement** – `RegexMatchStatement`

Regex pattern set match rule statement

The regex pattern set match inspects the part of the web request that you specify for the regular expression patterns that you've specified inside a regex pattern set.

> **Note**
> Each regex pattern set match rule references a regex pattern set, which you create and maintain independent of your rules. This allows you to use the single set in multiple rules. When you update the referenced regex pattern set, AWS WAF automatically updates all rules that reference it.
> For information about creating and managing a regex pattern set, see Creating and managing a regex pattern set (p. 87).

A regex pattern set match statement instructs AWS WAF to search for any of the patterns in the set inside the request component that you choose. A web request will match the pattern set rule statement if the request component matches any of the patterns in the set.

If you want to combine your regex pattern matches using logic, for example to match against some regular expressions and not match against others, consider using Regex match rule statement (p. 69).

Nestable – You can nest this statement type.

WCUs – 25 WCUs per regex pattern set, as a base cost. If you use the request component **All query parameters**, add 10 WCUs. If you use the request component **JSON body**, double the base cost WCUs. For each **Text transformation** that you apply, add 10 WCUs.

This statement type operates on a web request component, and requires the following request component settings:

- **Request components** – The part of the web request to inspect, for example, a query string or the body.

  > **Note**
  > If you use the request component **Body** or **JSON body**, only the first 8 KB (8,192 bytes) of the request body are forwarded to AWS WAF for inspection. AWS WAF does not support inspecting the entire contents of web requests whose bodies exceed 8 KB.

  For information about web request components, see Request component (p. 75).
- **Optional text transformations** – Transformations that you want AWS WAF to perform on the request component before inspecting it. For example, you could transform to lowercase or normalize white space. If you specify more than one transformation, AWS WAF processes them in the order listed. For information, see Text transformations (p. 78).

This statement requires the following settings:

- Regex pattern set specification – Choose the regex pattern set that you want to use from the list or create a new one.

Where to find this

- **Rule builder** on the console – For **Match type**, choose **String match condition** > **Matches pattern from regular expression set**.
- **API statement** – `RegexPatternSetReferenceStatement`

Rule group statement

The rule group rule statement adds a reference to your web ACL rules list to a rule group that you manage. You don't see this option under your rule statements on the console, but when you work with the JSON format of your web ACL, any of your own rule groups that you've added show up under the web ACL rules as this type. For information about using your own rule groups, see Managing your own rule groups (p. 50).

When you add a rule group to a web ACL, you can exclude individual rules in the group from running, and you can override the actions of all rules in the rule group, to count only. For more information, see Web ACL rule and rule group evaluation (p. 19).

Not nestable – You can't nest this statement type inside other statements, and you can't include it in a rule group. You can include it directly in a web ACL. You can narrow the scope of the requests that the rule group evaluates by adding one of the nestable statements within the rule group statement, as a scope-down statement.

WCUs – Set for the rule group at creation.

Where to find this

- **Console** – During the process of creating a web ACL, on the **Add rules and rule groups** page, choose **Add my own rules and rule groups**, **Rule group**, and then add the rule group that you want to use.
- **API statement** – `RuleGroupReferenceStatement`

Size constraint rule statement

A size constraint statement compares a number of bytes against the size of a request component, using a comparison 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.

> **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.

Nestable – You can nest this statement type.

WCUs – 1 WCU, as a base cost. If you use the request component **All query parameters**, add 10 WCUs. If you use the request component **JSON body**, double the base cost WCUs. For each **Text transformation** that you apply, add 10 WCUs.

This statement type operates on a web request component, and requires the following request component settings:

- **Request components** – The part of the web request to inspect, for example, a query string or the body.

  > **Note**
  > If you use the request component **Body** or **JSON body**, only the first 8 KB (8,192 bytes) of the request body are forwarded to AWS WAF for inspection. AWS WAF does not support inspecting the entire contents of web requests whose bodies exceed 8 KB.

For information about web request components, see Request component (p. 75).

- **Optional text transformations** – Transformations that you want AWS WAF to perform on the request component before inspecting it. For example, you could transform to lowercase or normalize white space. If you specify more than one transformation, AWS WAF processes them in the order listed. For information, see Text transformations (p. 78).

Additionally, this statement requires the following settings:

- **Size match condition** – This indicates the numerical comparison operator to use to compare the size that you provide with the request component that you've chosen. Choose the operator from the list.
- **Size** – The size setting, in bytes, to use in the comparison.

Where to find this

- **Rule builder** on the console – For **Match type**, under **Size match condition**, choose the condition that you want to use.
- **API statement** – `SizeConstraintStatement`

SQL injection attack rule statement

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. An 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.

Nestable – You can nest this statement type.

WCUs – 20 WCUs, as a base cost. If you use the request component **All query parameters**, add 10 WCUs. If you use the request component **JSON body**, double the base cost WCUs. For each **Text transformation** that you apply, add 10 WCUs.

This statement type operates on a web request component, and requires the following request component settings:

- **Request components** – The part of the web request to inspect, for example, a query string or the body.

  Note
  If you use the request component **Body** or **JSON body**, only the first 8 KB (8,192 bytes) of the request body are forwarded to AWS WAF for inspection. AWS WAF does not support inspecting the entire contents of web requests whose bodies exceed 8 KB.

  For information about web request components, see Request component (p. 75).
- **Optional text transformations** – Transformations that you want AWS WAF to perform on the request component before inspecting it. For example, you could transform to lowercase or normalize white space. If you specify more than one transformation, AWS WAF processes them in the order listed. For information, see Text transformations (p. 78).

Where to find this

- **Rule builder** on the console – For **Match type**, choose **Attack match conditions** > **Contains SQL injection attacks**.

- **API statement** – `SqliMatchStatement`

String match rule statement

A string match statement indicates the string that you want AWS WAF to search for in a request, where in the request to search, and how. For example, you can look for a specific string at the start of any query string in the request or as an exact match for the request's `User-agent` header. Usually, the string consists of printable ASCII characters, but you can use any character from hexadecimal 0x00 to 0xFF (decimal 0 to 255).

Nestable – You can nest this statement type.

WCUs – The base cost depends on the type of match that you use.

- **Exactly matches string** – 2
- **Starts with string** – 2
- **Ends with string** – 2
- **Contains string** – 10
- **Contains word** – 10

If you use the request component **All query parameters**, add 10 WCUs. If you use the request component **JSON body**, double the base cost WCUs. For each **Text transformation** that you apply, add 10 WCUs.

This statement type operates on a web request component, and requires the following request component settings:

- **Request components** – The part of the web request to inspect, for example, a query string or the body.

  Note
  If you use the request component **Body** or **JSON body**, only the first 8 KB (8,192 bytes) of the request body are forwarded to AWS WAF for inspection. AWS WAF does not support inspecting the entire contents of web requests whose bodies exceed 8 KB.

  For information about web request components, see Request component (p. 75).
- **Optional text transformations** – Transformations that you want AWS WAF to perform on the request component before inspecting it. For example, you could transform to lowercase or normalize white space. If you specify more than one transformation, AWS WAF processes them in the order listed. For information, see Text transformations (p. 78).

Additionally, this statement requires the following settings:

- **String to match** – This is the string that you want AWS WAF to compare to the specified request component. Usually, the string consists of printable ASCII characters, but you can use any character from hexadecimal 0x00 to 0xFF (decimal 0 to 255).
- **String match condition** – This indicates the search type that you want AWS WAF to perform.
  - **Exactly matches string** – The string and the value of the request component are identical.
  - **Starts with string** – The string appears at the beginning of the request component.
  - **Ends with string** – The string appears at the end of the request component.
  - **Contains string** – The string appears anywhere in the request component.
  - **Contains word** – The string that you specify must appear in the request component. For this option, the string that you specify must contain only alphanumeric characters or underscore (A-Z, a-z, 0-9, or _).

One of the following must be true for the request to match:

- The string exactly matches the value of the request component, such as the value of a header.
- The string is at the beginning of the request component and is followed by a character other than an alphanumeric character or underscore (_), for example, `BadBot;`.
- The string is at the end of the request component and is preceded by a character other than an alphanumeric character or underscore (_), for example, `;BadBot`.
- The string is in the middle of the request component and is preceded and followed by characters other than alphanumeric characters or underscore (_), for example, `-BadBot;`.

Where to find this

- **Rule builder** on the console – For **Match type**, choose **String match condition**, and then fill in the strings that you want to match against.
- **API statement** – `ByteMatchStatement`

Cross-site scripting attack rule statement

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 part of web requests, such as the URI or the query string, that you want AWS WAF to inspect for possible malicious scripts.

When you create cross-site scripting match conditions, you specify filters. The filters 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.

Nestable – You can nest this statement type.

WCUs – 40 WCUs, as a base cost. If you use the request component **All query parameters**, add 10 WCUs. If you use the request component **JSON body**, double the base cost WCUs. For each **Text transformation** that you apply, add 10 WCUs.

This statement type operates on a web request component, and requires the following request component settings:

- **Request components** – The part of the web request to inspect, for example, a query string or the body.

  Note
  If you use the request component **Body** or **JSON body**, only the first 8 KB (8,192 bytes) of the request body are forwarded to AWS WAF for inspection. AWS WAF does not support inspecting the entire contents of web requests whose bodies exceed 8 KB.

  For information about web request components, see Request component (p. 75).
- **Optional text transformations** – Transformations that you want AWS WAF to perform on the request component before inspecting it. For example, you could transform to lowercase or normalize white space. If you specify more than one transformation, AWS WAF processes them in the order listed. For information, see Text transformations (p. 78).

Where to find this

- **Rule builder** on the console – For **Match type**, choose **Attack match conditions** > **Contains XSS injection attacks**.

- **API statement** – `XssMatchStatement`

Web request component settings

This section describes the settings that you can specify for rule statements that inspect a component of the web request. For information on usage, see the individual rule statements.

Request component

The request component specifies the part of a web request for AWS WAF to inspect. You specify this for standard rule statements that look for patterns inside the web request. These include regex pattern match, SQL injection attack, and size constraint statements.

Unless otherwise noted, if a web request doesn't have the request component that's specified in the rule statement, the request results as not matching the rule.

> **Note**
> You specify a single request component for each rule statement that requires it. To inspect more than one component of a request, create a rule statement for each component.

The AWS WAF console and API documentation provide guidance for these settings in the following locations:

- **Rule builder** on the console – in the **Statement** settings for a regular rule type, choose the component that you want to inspect in the **Inspect** dialogue under **Request components**.
- **API statement contents** – `FieldToMatch`

Here are the options for the part of the web request to inspect:

Options for the part of the request to inspect

Header

A specific request header. For this option, you also choose the name of the header in the **Header type** field, for example, `User-Agent` or `Referer`.

HTTP method

The HTTP method, which indicates the type of operation that the web request is asking the origin to perform.

Query string

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

> **Note**
> For cross-site scripting match conditions, we recommend that you choose **All query parameters** instead of **Query string**. Choosing **All query parameters** adds 10 WCUs to the base cost.

Single query parameter

Any parameter that you have defined as part of the query string. AWS WAF inspects the value of the parameter that you specify.

For this option, you also specify a **Query parameter name**. For example, if the URL is `www.xyz.com?UserName=abc&SalesRegion=seattle`, you can specify `UserName` or `SalesRegion` for the name. The maximum length for the name is 30 characters. The name is

not case sensitive, so if you specify `UserName` as the name, AWS WAF matches all variations of `UserName`, including `username` and `UsERName`.

If the query string contains more than one instance of the name that you've specified, AWS WAF inspects all the values for a match, using `OR` logic. For example, in the URL `www.xyz.com?SalesRegion=boston&SalesRegion=seattle`, AWS WAF evaluates the name that you've specified against `boston` and `seattle`. If either is a match, the inspection is a match.

All query parameters

Similar to **Single query parameter**, but AWS WAF inspects the values of all parameters within the query string. For example, if the URL is `www.xyz.com?UserName=abc&SalesRegion=seattle`, AWS WAF triggers a match if either the value of `UserName` or `SalesRegion` match the inspection criteria.

Choosing this option adds 10 WCUs to the base cost.

URI path

The part of a URL that identifies a resource, for example, `/images/daily-ad.jpg`. If you don't use a text transformation with this option, AWS WAF doesn't normalize the URI and inspects it just as it receives it from the client in the request.

Body

The part of the request that immediately follows the request headers, evaluated as plain text. This contains any additional data that is needed for the web request, for example, data from a form.

- In the console, you select this under the **Request option** choice **Body**, by selecting the **Content type** choice **Plain text**.
- In the API, in the rule's `FieldToMatch` specification, you specify `Body` to inspect the request body as plain text.

  Note
  Only the first 8 KB (8,192 bytes) of the request body are forwarded to AWS WAF for inspection. If you don't need to inspect more than 8 KB, you can guarantee that you don't allow additional bytes in by combining your statement that inspects the body of the web request with a size constraint rule statement that enforces an 8 KB max size on the body of the request. For information about size constraint statements, see Size constraint rule statement (p. 71). AWS WAF doesn't support inspecting the entire contents of web requests whose bodies exceed 8 KB.

You can also evaluate the body as parsed JSON. For information about this, see the sections that follow.

JSON body

The part of the request that immediately follows the request headers, evaluated as parsed JSON. The body contains any additional data that is needed for the web request, for example, data from a form. You can also evaluate the body as plain text. For information about this, see the preceding section.

- In the console, you select this under the **Request option** choice **Body**, by selecting the **Content type** choice **JSON**.
- In the API, in the rule's `FieldToMatch` specification, you specify `JsonBody`.

  Note
  Only the first 8 KB (8,192 bytes) of the request body are forwarded to AWS WAF for inspection. If you don't need to inspect more than 8 KB, you can guarantee that you don't allow additional bytes in by combining your statement that inspects the body of the web request with a size constraint rule statement that enforces an 8 KB max size on the body of the request. For information about size constraint statements, see Size constraint rule statement (p. 71). AWS WAF doesn't support inspecting the entire contents of web requests whose bodies exceed 8 KB.

For details about JSON body inspection, see the following section. Choosing the JSON body option doubles the match statement's base cost WCUs. For example, if the match statement base cost is 5 WCUs without JSON parsing, using JSON parsing doubles the cost to 10 WCUs.

JSON body request component

JSON body inspection provides a specialized inspection of a web request body. For general information about web request body inspection, see the prior section.

Note
Only the first 8 KB (8,192 bytes) of the request body are forwarded to AWS WAF for inspection. If you don't need to inspect more than 8 KB, you can guarantee that you don't allow additional bytes in by combining your statement that inspects the body of the web request with a size constraint rule statement that enforces an 8 KB max size on the body of the request. For information about size constraint statements, see Size constraint rule statement (p. 71). AWS WAF doesn't support inspecting the entire contents of web requests whose bodies exceed 8 KB.

When AWS WAF inspects the web request body as parsed JSON, it parses and extracts the elements from the JSON and inspects the parts that you indicate using the rule's match statement criteria.

Choosing this option doubles the match statement's base cost WCUs. For example, if the match statement base cost is 5 WCUs without JSON parsing, using JSON parsing doubles the cost to 10 WCUs.

With this option, AWS WAF runs two match patterns against the web request body, with the output of the first used as input to the second:

1. AWS WAF parses and extracts the JSON content and identifies the elements to inspect. To do this, AWS WAF uses the criteria that you provide in the rule's JSON body specification.
2. AWS WAF applies any text transformations to the extracted elements and then matches the resulting JSON element set against the rule statement's match criteria. If any of the elements match, the web request is a match for the rule.

You specify the following criteria for AWS WAF to use for the first pattern matching step, to identify the JSON elements to inspect:

- **Body parsing fallback behavior** – What AWS WAF should do if it fails to completely parse the JSON body. The options are the following:
  - **None (default behavior)** - AWS WAF evaluates the content only up to the point where it encountered a parsing error.
  - **Evaluate as string** - Inspect the body as plain text. AWS WAF applies the text transformations and inspection criteria that you defined for the JSON inspection to the body text string.
  - **Match** - Treat the web request as matching the rule statement. AWS WAF applies the rule action to the request.
  - **No match** - Treat the web request as not matching the rule statement.

  AWS WAF does its best to parse the entire JSON body, but might be forced to stop for reasons such as invalid characters, duplicate keys, truncation, and any content whose root node isn't an object or an array.

  AWS WAF parses the JSON in the following examples as two valid key, value pairs:
  - Missing comma: `{"key1":"value1""key2":"value2"}`
  - Missing colon: `{"key1":"value1","key2""value2"}`
  - Extra colons: `{"key1"::"value1","key2""value2"}`
- **JSON match scope** – The types of elements in the JSON that AWS WAF should inspect. You can specify **Keys**, **Values**, or **All** for both keys and values.

- **Content to inspect** – The elements in the parsed and extracted JSON that you want AWS WAF to inspect.

  You must specify one of the following:

  - **Full JSON content** - Evaluate all elements in the parsed JSON.
  - **Only included elements** - Evaluate only elements in the JSON that match the JSON Pointer criteria that you provide. For information about the JSON Pointer syntax, see the Internet Engineering Task Force (IETF) documentation JavaScript Object Notation (JSON) Pointer.

    Don't use this option to include *all* paths in the JSON. Use **Full JSON content** instead.

    For example, in the console, you can provide the following:

    ```
    /dogs/0/name
    /dogs/1/name
    ```

    In the API or CLI, you can provide the following:

    ```
    "IncludedPaths": ["/dogs/0/name", "/dogs/1/name"]
    ```

Example JSON body inspection scenario

If the included elements setting is `/a/b`, then for the following JSON body:

```
{
  "a":{
    "c":"d",
    "b":{
      "e":{
        "f":"g"
      }
    }
  }
}
```

The following list describes what AWS WAF would evaluate for each match scope setting. The key `b`, which is part of the included elements path, isn't evaluated.

- For a match scope set to all: `e`, `f`, and `g`.
- For a match scope set to keys: `e` and `f`.
- For a match scope set to values: `g`.

Text transformations

In statements that look for patterns or set constraints, you can provide transformations for AWS WAF to apply before inspecting the request. A transformation reformats a web request to eliminate some of the unusual formatting that attackers use in an effort to bypass AWS WAF.

When you use this with the JSON body request component selection, AWS WAF applies your transformations after parsing and extracting the elements to inspect from the JSON. For more information, see the prior section, JSON body request component (p. 77).

If you provide more than one transformation, you also set the order for AWS WAF to apply them.

WCUs – Each text transformation is 10 WCUs.

The AWS WAF console and API documentation also provide guidance for these settings in the following locations:

- **Rule builder** on the console – **Text transformation**. This option is available when you use request components.
- **API statement contents** – `TextTransformations`

Options for text transformations

Base64 decode

AWS WAF decodes a Base64-encoded string.

Base64 decode ext

AWS WAF decodes a Base64-encoded string, but uses a forgiving implementation that ignores characters that aren't valid.

Command line

This option mitigates situations where attackers might be injecting an operating system command line command and are 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`

Compress white space

AWS WAF replaces multiple spaces with one space and 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

CSS decode

AWS WAF decodes characters that were encoded using CSS 2.x escape rules `syndata.html#characters`. This function uses up to two bytes in the decoding process, so it can help to uncover ASCII characters that were encoded using CSS encoding that wouldn't typically be encoded. It's also useful in countering evasion, which is a combination of a backslash and non-hexadecimal characters. For example, ja\vascript for javascript.

Escape sequence decode

AWS WAF decodes the following ANSI C escape sequences: `\a, \b, \f, \n, \r, \t, \v, \ \, \?, \', \", \xHH (hexadecimal), \0OOO (octal)`. Encodings that aren't valid remain in the output.

Hex decode

AWS WAF decodes a string of hexadecimal characters into a binary.

HTML entity decode

AWS WAF replaces HTML-encoded characters with unencoded characters:

- Replaces `&quot;` with "
- Replaces ` ` with a non-breaking space, decimal 160
- Replaces `<` with <
- Replaces `>` 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

JS decode

AWS WAF decodes JavaScript escape sequences. If a `\uHHHH` code is in the full-width ASCII code range of `FF01-FF5E`, then the higher byte is used to detect and adjust the lower byte. If not, only the lower byte is used and the higher byte is zeroed, causing a possible loss of information.

Lowercase

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

MD5

AWS WAF calculates an MD5 hash from the data in the input. The computed hash is in a raw binary form.

None

AWS WAF inspects the web request as received, without any text transformations.

Normalize path

AWS WAF removes multiple slashes, directory self-references, and directory back-references that are not at the beginning of the input from an input string.

Normalize path win

AWS WAF processes this like `NORMALIZE_PATH`, but first converts backslash characters to forward slashes.

Remove nulls

AWS WAF removes all `NULL` bytes from the input.

Replace comments

AWS WAF replaces each occurrence of a C-style comment (/* ... */) with a single space. Multiple consecutive occurrences are not compressed. Unterminated comments are also replaced with a space (ASCII 0x20). However, a standalone termination of a comment (*/) is not acted upon.

Replace nulls

AWS WAF replaces `NULL` bytes in the input with space characters (ASCII 0x20).

SQL hex decode

AWS WAF decodes SQL hex data. For example, (`0x414243`) is decoded to (`ABC`).

URL decode

AWS WAF decodes a URL-encoded value.

URL decode uni

Like `URL_DECODE`, but with support for Microsoft-specific `%u` encoding. If the code is in the full-width ASCII code range of `FF01-FF5E`, the higher byte is used to detect and adjust the lower byte. Otherwise, only the lower byte is used and the higher byte is zeroed.

UTF8 to Unicode

AWS WAF converts all UTF-8 character sequences to Unicode. This helps input normalization, and minimizes false-positives and false-negatives for non-English languages.

Statements that reference a set or a rule group

Some rules use entities that are reusable and that are managed outside of your web ACLs, either by you, AWS, or an AWS Marketplace seller. When the reusable entity is updated, AWS WAF propagates the update to your rule. For example, if you use an AWS Managed Rules rule group in a web ACL, when AWS updates the rule group, AWS propagates the change to your web ACL, to update its behavior. If you use an IP set statement in a rule, when you update the set, AWS WAF propagates the change to all rules that reference it, so any web ACLs that use those rules are kept up-to-date with your changes.

The following are the reusable entities that you can use in a rule statement.

- **IP sets** – You create and manage your own IP sets. On the console, you can access these from the navigation pane. For information about managing IP sets, see IP sets and regex pattern sets (p. 84).
- **Regex match sets** – You create and manage your own regex match sets. On the console, you can access these from the navigation pane. For information about managing regex pattern sets, see IP sets and regex pattern sets (p. 84).
- **AWS Managed Rules rule groups** – AWS manages these rule groups. On the console, these are available for your use when you add a managed rule group to your web ACL. For more information about these, see AWS Managed Rules rule groups list (p. 38).
- **AWS Marketplace managed rule groups** – AWS Marketplace sellers manage these rule groups and you can subscribe to them to use them. To manage your subscriptions, on the navigation pane of the console, choose **AWS Marketplace**. The AWS Marketplace managed rule groups are listed when you add a managed rule group to your web ACL. For rule groups that you haven't yet subscribed to, you can find a link to AWS Marketplace on that page as well. For more information about AWS Marketplace seller managed rule groups, see AWS Marketplace managed rule groups (p. 49).
- **Your own rule groups** – You manage your own rule groups, usually when you need some behavior that isn't available through the managed rule groups. On the console, you can access these from the navigation pane. For more information, see Managing your own rule groups (p. 50).

Deleting a referenced set or rule group

When you delete a referenced entity, AWS WAF checks to see if it's currently being used in a web ACL. If AWS WAF finds that it's in use, it warns you. AWS WAF is almost always able to determine if an entity is being referenced by a web ACL. However, in rare cases, it might not be able to do so. If you need to be sure that the entity that you want to delete isn't in use, check for it in your web ACLs before deleting it.

Scope-down statements

You can add a scope-down statement inside some rules. The scope-down statement narrows the scope of the requests that the rule evaluates. If a rule has a scope-down statement, traffic is first evaluated using the scope-down statement. If it matches that, then it's evaluated using the rule's standard criteria. Traffic that doesn't match the scope-down statement results as not matching the rule. AWS WAF performs no further evaluation.

You can define a scope-down statement inside the following statement types:

- **Managed rule group statement** – If you add a scope-down statement to a managed rule group statement, any request that doesn't match the scope-down statement results as not matching the rule group. Only those requests that match the scope-down statement are evaluated against the rule group. For managed rule groups with pricing that's based on the number of requests evaluated, scope-

down statements can help contain costs. For more information about managed rule group statements, see Managed rule group statement (p. 66).

- **Rate-based rule statement** – A rate-based rule without a scope-down statement controls the rate of all requests that come in to your applications. If you want to only control the rate for a specific category of requests, you add a scope-down statement to the rate-based rule. For example, to only track and control the rate of requests from a specific geographical area, you specify that geographical ares in a geographic match rule as the scope-down statement. For more information about rate-based rule statements, see Rate-based rule statement (p. 68).

You can use any nestable rule in a scope-down statement. The WCUs for the scope-down statement are calculated as the WCUs required for the rule statements that you use in it. For a list of available statements, see Rule statements list (p. 61).

Forwarded IP address

This section applies to rule statements that use the IP address of a web request. By default, AWS WAF uses the IP address from the web request origin. However, if a web request goes through one or more proxies or load balancers, the web request origin will contain the address of the last proxy, and not the originating address of the client. In this case, the originating client address is usually forwarded in another HTTP header. This header is typically `X-Forwarded-For` (XFF), but it can be a different one.

Rule statements that use IP addresses

The rule statements that use IP addresses are the following:

- IP set match (p. 65) - Inspects the IP address for a match with the addresses that are defined in an IP set.
- Geographic match (p. 64) - Uses the IP address to determine country of origin and matches that against a list of countries.
- Rate-based (p. 68) - Aggregates requests by their IP addresses to ensure that no individual IP address sends requests at too high a rate.

You can instruct AWS WAF to use a forwarded IP address for any of these rule statements, either from the `X-Forwarded-For` header or from another HTTP header, instead of using the web request's origin. For details on how to provide the specifications, see the guidance for the individual rule statement types.

IP addresses used in AWS WAF Bot Control

The Bot Control managed rule group verifies bots using the IP addresses from AWS WAF. If you use Bot Control and you have verified bots that route through a proxy or load balancer, you need to explicitly allow them using a custom rule. For example, you can configure a custom IP set match rule that uses forwarded IP addresses to detect and allow your verified bots. You can use the rule to customize your bot management in a number of ways. For information and examples, see AWS WAF Bot Control (p. 94).

General considerations for using forwarded IP addresses

Before you use a forwarded IP address, note the following general caveats:

- A header can be modified by proxies along the way, and the proxies might handle the header in different ways.
- Attackers might alter the contents of the header in an attempt to bypass AWS WAF inspections.
- The IP address inside the header can be malformed or invalid.
- The header that you specify might not be present at all in a request.

Considerations for using forwarded IP addresses with AWS WAF

The following list describes requirements and caveats for using forwarded IP addresses in AWS WAF:

- For any single rule, you can specify one header for the forwarded IP address. The header specification is case insensitive.
- For rate-based rule statements, any nested scoping statements do not inherit the forwarded IP configuration. Specify the configuration for each statement that uses a forwarded IP address.
- For geo match and rate-based rules, AWS WAF uses the first address in the header. For example, if a header contains “10.1.1.1, 127.0.0.0, 10.10.10.10”, AWS WAF uses “10.1.1.1”.
- For IP set match, you indicate whether to match against the first, last, or any address in the header. If you specify any, AWS WAF inspects all addresses in the header for a match, up to 10 addresses. If the header contains more than 10 addresses, AWS WAF inspects the last 10.
- Headers that contain multiple addresses must use a comma separator between the addresses. If a request uses a separator other than a comma, AWS WAF considers the IP addresses in the header malformed.
- If the IP addresses inside the header are malformed or invalid, AWS WAF designates the web request as matching the rule or not matching, according to the fallback behavior that you specify in the forwarded IP configuration.
- If the header that you specify isn’t present in a request, AWS WAF doesn’t apply the rule to the request at all. This means that AWS WAF doesn't apply the rule action and doesn't apply the fallback behavior.
- A rule statement that uses a forwarded IP header for the IP address won’t use the IP address that’s reported by the web request origin.

Best practices for using forwarded IP addresses with AWS WAF

When you use forwarded IP addresses, use the following best practices:

- Carefully consider all possible states of your request headers before enabling forwarded IP configuration. You might need to use more than one rule to get the behavior you want.
- To inspect multiple forwarded IP headers or to inspect the web request origin and a forwarded IP header, use one rule for each IP address source.
- To block web requests that have an invalid header, set the rule action to block and set the fallback behavior for the forwarded IP configuration to match.

Example JSON for forwarded IP addresses

The following geo match statement matches only if the `X-Forwarded-For` header contains an IP whose country of origin is `US`:

```
{
  "Name": "XFFTestGeo",
  "Priority": 0,
  "Action": {
    "Block": {}
  },
  "VisibilityConfig": {
    "SampledRequestsEnabled": true,
    "CloudWatchMetricsEnabled": true,
    "MetricName": "XFFTestGeo"
  },
  "Statement": {
    "GeoMatchStatement": {
      "CountryCodes": [
        "US"
      ],
      "ForwardedIPConfig": {
        "HeaderName": "x-forwarded-for",
```

```
        "FallbackBehavior": "MATCH"
      }
    }
  }
}
```

The following rate-based rule aggregates requests based on the first IP in the `X-Forwarded-For` header. The rule counts only requests the match the nested geo match statement. The nested geo match statement also uses the `X-Forwarded-For` header to determine whether the IP address indicates a country of origin of `US`. If it does, or if the header is present but malformed, the geo match statement returns a match.

```
{
  "Name": "XFFTestRateGeo",
  "Priority": 0,
  "Action": {
    "Block": {}
  },
  "VisibilityConfig": {
    "SampledRequestsEnabled": true,
    "CloudWatchMetricsEnabled": true,
    "MetricName": "XFFTestRateGeo"
  },
  "Statement": {
    "RateBasedStatement": {
      "Limit": "100",
      "AggregateKeyType": "FORWARDED_IP",
      "ScopeDownStatement": {
        "GeoMatchStatement": {
          "CountryCodes": [
            "US"
          ],
          "ForwardedIPConfig": {
            "HeaderName": "x-forwarded-for",
            "FallbackBehavior": "MATCH"
          }
        }
      },
      "ForwardedIPConfig": {
        "HeaderName": "x-forwarded-for",
        "FallbackBehavior": "MATCH"
      }
    }
  }
}
```

IP sets and regex pattern sets

AWS WAF stores some more complex information in sets that you use by referencing them in your rules. Each of these sets has a name and is assigned an Amazon Resource Name (ARN) at creation. You can manage these sets from inside your rule statements and you can access and manage them on their own, through the console navigation pane.

Eventual consistency

When you make changes to web ACLs or web ACL components, like rules and rule groups, AWS WAF propagates the changes everywhere that the web ACL and its components are stored and used. Your changes are applied within seconds, but there might be a brief period of inconsistency when the changes have arrived in some places and not in others. So, for example, if you add an IP address to an IP set that's referenced by a blocking rule in a web ACL, the new address might briefly be blocked in one area while

still allowed in another. This temporary inconsistency can occur when you first associate a web ACL with an AWS resource and when you change a web ACL that is already associated with a resource. Generally, any inconsistencies of this type last only a few seconds.

Topics

Creating and managing an IP set

An IP set provides a collection of IP addresses and IP address ranges that you want to use together in a rule statement. IP sets are AWS resources.

To use an IP set in a web ACL or rule group, you first create an AWS resource, `IPSet` with your address specifications. Then you reference the set when you add an IP set rule statement to a web ACL or rule group.

Topics

Creating an IP set

Follow the procedure in this section to create a new IP set.

> **Note**
> In addition to the procedure in this section, you have the option to add a new IP set when you add an IP match rule to your web ACL or rule group. Choosing that option requires you to provide the same settings as those required by this procedure.

To create an IP set

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **IP sets** and then **Create IP set**.
3. Enter a name and description for the IP set. You'll use these to identify the set when you want to use it.

   > **Note**
   > You can't change the name after you create the IP set.
4. For **Region**, choose the Region where you want to store the IP set. To use an IP set in web ACLs that protect Amazon CloudFront distributions, you must use Global (CloudFront).
5. For **IP version**, select the version you want to use.
6. In the **IP addresses** text box, enter one IP address or IP address range per line, in CIDR notation. AWS WAF supports all IPv4 and IPv6 CIDR ranges except for `/0`. For more information about CIDR notation, see the Wikipedia article Classless Inter-Domain Routing.

   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, enter **2620:0:2d0:200::/64**.

7. Review the settings for the IP set, and choose **Create IP set**.

Using an IP set in a rule group or Web ACL

To use an IP set, add a rule statement that references it to the rule group or web ACL where you need it. For information, see IP set match rule statement (p. 65).

Editing an IP set

To add or remove IP addresses or IP address ranges from an IP set or change its description, perform the following procedure.

To edit an IP set

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **IP sets**.
3. Select the IP set that you want to edit and choose **Edit**.
4. Modify the IP version and addresses as needed. In the **IP addresses** text box, you must have one IP address or IP address range per line, in CIDR notation. AWS WAF supports all IPv4 and IPv6 CIDR ranges except for `/0`. For more information about CIDR notation, see the Wikipedia article Classless Inter-Domain Routing. For addresses, enter one IP address or IP address range per line, in 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, enter **2620:0:2d0:200::/64**.

5. Choose **Save changes**.

Deleting an IP set

Follow the guidance in this section to delete a referenced set.

Deleting referenced sets and rule groups

When you delete an entity that you can use in a web ACL, like an IP set, regex pattern set, or rule group, AWS WAF checks to see if the entity is currently being used in a web ACL. If it finds that it is in use, AWS WAF warns you. AWS WAF is almost always able to determine if an entity is being referenced by a web ACL. However, in rare cases it might not be able to do so. If you need to be sure that nothing is currently using the entity, check for it in your web ACLs before deleting it. If the entity is a referenced set, also check that no rule groups are using it.

To delete an IP set

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **IP sets**.
3. Select the IP set that you want to delete and choose **Delete**.

Creating and managing a regex pattern set

A regex pattern set provides a collection of regular expressions that you want to use together in a rule statement. Regex pattern sets are AWS resources.

To use a regex pattern set in a web ACL or rule group, you first create an AWS resource, `RegexPatternSet` with your regex pattern specifications. Then you reference the set when you add a regex pattern set rule statement to a web ACL or rule group. A regex pattern set must contain at least one regex pattern.

If your regex pattern set contains more than one regex pattern, when it's used in a rule, the pattern matching is combined with `OR` logic. That is, a web request will match the pattern set rule statement if the request component matches any of the patterns in the set.

AWS WAF supports the pattern syntax used by the PCRE library `libpcre`. The library is documented at PCRE - Perl Compatible Regular Expressions.

Regex pattern use limitations

AWS WAF doesn't support all contructs of the library. For example, it supports some zero-width assertions, but not all. We do not have comprehensive list of the constructs that are supported. However, if you provide a regex pattern that isn't valid or use unsupported constructs, the AWS WAF API reports a failure.

AWS WAF does not support the following PCRE patterns:

- Backreferences and capturing subexpressions
- Subroutine references and recursive patterns
- Conditional patterns
- Backtracking control verbs
- The \C single-byte directive
- The \R newline match directive
- The \K start of match reset directive
- Callouts and embedded code
- Atomic grouping and possessive quantifiers

Topics

Creating a regex pattern set

Follow the procedure in this section to create a new regex pattern set.

To create a regex pattern set

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **Regex pattern sets** and then **Create regex pattern set**.
3. Enter a name and description for the regex pattern set. You'll use these to identify it when you want to use the set.

Note
You can't change the name after you create the regex pattern set.

4. For **Region**, choose the Region where you want to store the regex pattern set. To use a regex pattern set in web ACLs that protect Amazon CloudFront distributions, you must use Global (CloudFront).
5. In the **Regular expressions** text box, enter one regex pattern per line.

   For example, the regular expression `I[a@]mAB[a@]dRequest` matches the following strings: `IamABadRequest`, `IamAB@dRequest`, `I@mABadRequest`, and `I@mAB@dRequest`.

   AWS WAF supports the pattern syntax used by the PCRE library `libpcre`. The library is documented at PCRE - Perl Compatible Regular Expressions.

   AWS WAF doesn't support all contructs of the library. For example, it supports some zero-width assertions, but not all. We do not have comprehensive list of the constructs that are supported. However, if you provide a regex pattern that isn't valid or use unsupported constructs, the AWS WAF API reports a failure.

   AWS WAF does not support the following PCRE patterns:

   - Backreferences and capturing subexpressions
   - Subroutine references and recursive patterns
   - Conditional patterns
   - Backtracking control verbs
   - The \C single-byte directive
   - The \R newline match directive
   - The \K start of match reset directive
   - Callouts and embedded code
   - Atomic grouping and possessive quantifiers
6. Review the settings for the regex pattern set, and choose **Create regex pattern set**.

Using a regex pattern set in a rule group or Web ACL

To use a regex pattern set in a rule group or web ACL, in the console, when you add or update the rules in your rule group or web ACL, in the **Rule builder** interface, for **Request option**, choose the request component that you want to compare to your pattern set. Choose **Match type** > **String match condition** > **Matches pattern from regular expression**, and then choose the name of the regex pattern set that you want to use.

Deleting a regex pattern set

Follow the guidance in this section to delete a referenced set.

Deleting referenced sets and rule groups

When you delete an entity that you can use in a web ACL, like an IP set, regex pattern set, or rule group, AWS WAF checks to see if the entity is currently being used in a web ACL. If it finds that it is in use, AWS WAF warns you. AWS WAF is almost always able to determine if an entity is being referenced by a web ACL. However, in rare cases it might not be able to do so. If you need to be sure that nothing is currently using the entity, check for it in your web ACLs before deleting it. If the entity is a referenced set, also check that no rule groups are using it.

To delete a regex pattern set

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.

2. In the navigation pane, choose **Regex pattern sets**.
3. Select the regex pattern set that you want to delete and choose **Delete**.

Customizing web requests and responses in AWS WAF

You can add custom web request and response handling behavior to your AWS WAF rule actions and default web ACL actions. Your custom settings apply whenever the action they're attached to applies.

You can customize web requests and responses in the following ways:

- With allow and count actions, you can insert custom headers into the web request. When AWS WAF forwards the web request to the protected resource, the request contains the entire original request plus the custom headers that you've inserted.

- With block actions, you can define a complete custom response, with response code, headers, and body. The protected resource responds to the request using the custom response provided by AWS WAF. Your custom response replaces the default block action response of `403 (Forbidden)`.

Action settings that you can customize

You can specify a custom request or response when you define the following action settings:

- Rule action. For information, see AWS WAF rule action (p. 53).
- Default action for a web ACL. For information, see Deciding on the default action for a web ACL (p. 21).

Action settings that you cannot customize

You *cannot* specify custom request handling in the override action for a rule group that you use in a web ACL. See Web ACL rule and rule group evaluation (p. 19). Also see Managed rule group statement (p. 66) and Rule group statement (p. 71).

Eventual consistency

When you make changes to web ACLs or web ACL components, like rules and rule groups, AWS WAF propagates the changes everywhere that the web ACL and its components are stored and used. Your changes are applied within seconds, but there might be a brief period of inconsistency when the changes have arrived in some places and not in others. So, for example, if you add an IP address to an IP set that's referenced by a blocking rule in a web ACL, the new address might briefly be blocked in one area while still allowed in another. This temporary inconsistency can occur when you first associate a web ACL with an AWS resource and when you change a web ACL that is already associated with a resource. Generally, any inconsistencies of this type last only a few seconds.

Maximum settings for custom request and response handling

For information about maximum settings for custom request and response handling, see AWS WAF quotas (p. 135).

Topics

Custom request header insertions for allow and count actions

You can instruct AWS WAF to insert custom headers into the original HTTP request for rule actions or web ACL default actions that are set to allow or count. You can only add to the request. You can't modify or replace any part of the original request. For more information about rule actions, see AWS WAF rule action (p. 53). For more information about default web ACL actions, see Deciding on the default action for a web ACL (p. 21).

Use cases for custom header insertion include signaling a downstream application to process the request differently based on the inserted headers, and flagging the request for analysis.

Custom request header names

AWS WAF prefixes all request headers that it inserts with `x-amzn-waf-`, to avoid confusion with the headers that are already in the request. For example, if you specify the header name `sample`, AWS WAF inserts the header `x-amzn-waf-sample`.

Headers with the same name

If the request already has a header with the same name that AWS WAF is inserting, AWS WAF overwrites the header. So, if you define headers in multiple rules with identical names, the last rule to inspect the request and find a match would have its header added, and any previous rules would not.

Custom headers with the count action

The count rule action doesn't stop the processing of the web request. If you insert custom headers with a rule that uses the count action, subsequent rules might also insert custom headers. For information about the count rule action behavior, see AWS WAF rule action (p. 53).

For example, suppose you have the following rules, prioritized in the order shown:

1. RuleA with a count action and a customized header named `RuleAHeader`.
2. RuleB with an allow action and a customized header named `RuleBHeader`.

If a request matches both RuleA and RuleB, AWS WAF inserts the headers `x-amzn-waf-RuleAHeader` and `x-amzn-waf-RuleBHeader`, and then forwards the request to the protected resource.

AWS WAF inserts custom headers into a web request when it finishes inspecting the request. So if you use custom request handling with a rule that has the action set to count, the custom headers that you add are not inspected by subsequent rules.

Example custom request handling

You define custom request handling for a rule's action or for a web ACL's default action. The following listing shows the JSON for custom handling added to the default action for a web ACL.

```
{
 "Name": "SampleWebACL",
 "Scope": "REGIONAL",
 "DefaultAction": {
  "Allow": {
   "CustomRequestHandling": {
    "InsertHeaders": [
     {
      "Name": "fruit",
      "Value": "watermelon"
     },
     {
      "Name": "pie",
```

```
      "Value": "apple"
     }
    ]
   }
  }
 },
 "Description": "Sample web ACL with custom request handling configured for default
 action.",
 "Rules": [],
 "VisibilityConfig": {
  "SampledRequestsEnabled": true,
  "CloudWatchMetricsEnabled": true,
  "MetricName": "SampleWebACL"
 }
}
```

Custom responses for block actions

You can instruct AWS WAF to send a custom HTTP response back to the client for rule actions or web ACL default actions that are set to block the request. For more information about rule actions, see AWS WAF rule action (p. 53). For more information about default web ACL actions, see Deciding on the default action for a web ACL (p. 21).

The AWS WAF block action defines the response that the associated protected resource sends back to the client. By default, the response is `403 (Forbidden)`. For more information about the block action, see AWS WAF rule action (p. 53).

When you define custom response handling for a block action, you define the status code, headers, and response body. For a list of status codes that you can use with AWS WAF, see the section that follows, Supported status codes for custom response (p. 93).

Use cases

The use cases for custom responses include the following:

- Sending a non-default status code back to the client.
- Sending a static error page back to the client.
- Redirecting the client to a different URL by specifying one of the `3xx` redirection status codes, like `301 (Moved Permanently)` or `302 (Found)`, and then specify a new header called `Location` with the new URL.

Interaction with responses that you define for your protected resources

Custom responses that you specify for the AWS WAF block action take precedence over any response specifications that you define in your protected resource.

The host service for the AWS resource that you protect with AWS WAF might allow custom response handling for web requests. Examples include the following:

- Amazon CloudFront allows you to customize the error page based on status code. For information, see Creating a Custom Error Page for Specific HTTP Status Codes in the *Amazon CloudFront Developer Guide*.
- Amazon API Gateway allows you to define the response and status code for your gateway. For information, see Gateway responses in API Gateway in the *Amazon API Gateway Developer Guide*.

It's not possible to combine AWS WAF custom responses with any response settings in the protected AWS resource. Control over the response belongs either completely with AWS WAF or completely with the protected resource:

1. If AWS WAF blocks a web request, AWS WAF determines the response that the protected resource sends back to the client. For the default block action, the response is `403 (Forbidden)`. If you customize the block action, then you provide the response entirely in your custom settings. These block action responses take precedence over any response settings that you might have defined in the protected resource itself.
2. If AWS WAF allows a web request to go through, then your configuration of the protected resource determines the response that it sends back to the client. For allowed requests, the only customization that you can configure in AWS WAF is the insertion of custom headers into the original request, before forwarding to the protected resource. This option is described in the preceding section, Custom request header insertions for allow and count actions (p. 90).

Custom response bodies

You define the body of a custom response within the context of the web ACL or rule group where you want to use it. After you've defined a custom response body, you can use it by reference anywhere else in the web ACL or rule group where you created it. In the individual block action settings, you reference the custom body that you want to use and you define the status code and header of the custom response.

When you create a custom response in the console, you can choose from response bodies that you've already defined or you can create a new body. Outside of the console, you define your custom response bodies at the web ACL or rule group level, and then reference them from the action settings within the web ACL or rule group. This is shown in the example JSON in the following section.

Custom response example

The following example lists the JSON for a rule group with custom response settings. The custom response body is defined for the entire rule group, then referenced by key in the rule action.

```
{
 "ARN": "test_rulegroup_arn",
 "Capacity": 1,

 "CustomResponseBodies": {
  "CustomResponseBodyKey1": {
   "Content": "This is a plain text response body.",
   "ContentType": "TEXT_PLAIN"
  }
 },

 "Description": "This is a test rule group.",
 "Id": "test_rulegroup_id",
 "Name": "TestRuleGroup",

 "Rules": [
  {
   "Action": {
    "Block": {
     "CustomResponse": {
      "CustomResponseBodyKey": "CustomResponseBodyKey1",
      "ResponseCode": 404,
      "ResponseHeaders": [
       {
        "Name": "BlockActionHeader1Name",
        "Value": "BlockActionHeader1Value"
       }
      ]
     }
    }
   },
   "Name": "GeoMatchRule",
   "Priority": 1,
```

```
   "Statement": {
    "GeoMatchStatement": {
     "CountryCodes": [
      "US"
     ]
    }
   },
   "VisibilityConfig": {
    "CloudWatchMetricsEnabled": true,
    "MetricName": "TestRuleGroupReferenceMetric",
    "SampledRequestsEnabled": true
   }
  }
 ],
 "VisibilityConfig": {
  "CloudWatchMetricsEnabled": true,
  "MetricName": "TestRuleGroupMetric",
  "SampledRequestsEnabled": true
 }
}
```

Supported status codes for custom response

For detailed information about HTTP status codes, see Hypertext Transfer Protocol Status Code Definitions and List of HTTP status codes.

The following are the HTTP status codes that AWS WAF supports for custom responses.

- 2xx Successful
  - 200 – OK
  - 201 – Created
  - 202 – Accepted
  - 204 – No Content
  - 206 – Partial Content
- 3xx Redirection
  - 300 – Multiple Choices
  - 301 – Moved Permanently
  - 302 – Found
  - 303 – See Other
  - 304 – Not Modified
  - 307 – Temporary Redirect
  - 308 – Permanent Redirect
- 4xx Client Error
  - 400 – Bad Request
  - 401 – Unauthorized
  - 403 – Forbidden
  - 404 – Not Found
  - 405 – Method Not Allowed
  - 408 – Request Timeout
  - 409 – Conflict
  - 411 – Length Required
  - 412 – Precondition Failed
  - 413 – Request Entity Too Large

  - 414 – Request-URI Too Long
  - 415 – Unsupported Media Type
  - 416 – Requested Range Not Satisfiable
  - 421 – Misdirected Request
  - 429 – Too Many Requests
- 5xx Server Error
  - 500 – Internal Server Error
  - 501 – Not Implemented
  - 502 – Bad Gateway
  - 503 – Service Unavailable
  - 504 – Gateway Timeout
  - 505 – HTTP Version Not Supported

AWS WAF Bot Control

Bot Control helps you manage bot activity to your site by categorizing and identifying common bots, verifying generally desirable bots, and detecting high confidence signatures of bots. Bot Control combines an AWS managed rule group with AWS WAF features that allow you to customize handling of your bot-related traffic. Bot Control primarily targets self-identifying, non-targeted bots, in order to give you the ability to monitor and control this category of bot traffic.

Bot Control is a managed rule group that gives you visibility and control over common and pervasive bot traffic to your applications. With Bot Control, you can easily monitor, block, or rate-limit bots such as scrapers, scanners, and crawlers. You can also allow common bots like status monitors and search engines. You can protect your applications using the Bot Control managed rule group alone, or with other AWS Managed Rules rule groups and your own custom AWS WAF rules.

Bot Control includes a console dashboard that shows how much of your current traffic is coming from bots, based on request sampling. With the Bot Control managed rule group added to your web ACL, you can take action against bot traffic and receive detailed, real-time information about common bot traffic coming to your applications.

When AWS WAF evaluates a web request against the Bot Control managed rule group, the evaluation adds labels to requests that it detects as bot related. The labels provide information, for example the category and name of the bot, which you can match against in your own custom AWS WAF rules.

The labels that are generated by the Bot Control managed rule group are included in Amazon CloudWatch metrics and your web ACL logs. You can use AWS Firewall Manager AWS WAF policies to deploy the Bot Control managed rule group across your applications in multiple accounts that are part of your organization in AWS Organizations.

Bot Control components

The main components of a Bot Control implementation are the following:

- **AWSManagedRulesBotControlRuleSet** – The Bot Control managed rule group whose rules detect and handle various categories of bots. For information about the rule group's rules, see AWS WAF Bot Control rule group (p. 45). You include this rule group in your web ACL using a managed rule group reference statement. This rule group add labels to web requests that it detects as bot traffic. You are charged additional fees when you use this rule group. For more information, see AWS WAF Pricing.
- **Bot Control dashboard** – The bot monitoring dashboard for your web ACL, available through the web ACL Bot Control tab. Use this dashboard to monitor your traffic and understand how much of it comes from various types of bots. This can be a starting point for customizing your bot management, as

described in this topic. You can also use it to verify your changes and monitor activity for various bots and bot categories.

- **Logging and metrics** – You can monitor your bot traffic and understand how the Bot Control managed rule group evaluates and handles it by configuring and enabling logs and Amazon CloudWatch metrics for your web ACL. The labels that Bot Control adds to your web requests are included in the logs and in Amazon CloudWatch metrics. For information about logging and metrics, see Logging web ACL traffic information (p. 107) and Monitoring with Amazon CloudWatch (p. 370).

  Depending on your needs and the traffic that you see, you might want to customize your Bot Control implementation. For example, you might want to exclude some traffic from Bot Control evaluation, or you might want to alter how it handles some of the bot traffic that it identifies, using AWS WAF features like scope-down statements or label matching rules.
- **Scope-down statements** – You can limit the scope of the web requests that the Bot Control managed rule group evaluates by adding a scope-down statement inside the Bot Control managed rule group reference statement. A scope-down statement can be any nestable rule statement. Traffic that doesn't match the scope-down statement results as not matching the rule group, and isn't evaluated by the Bot Control managed rule group. For more information about scope-down statements, see Scope-down statements (p. 81).

  Pricing for the Bot Control managed rule group is based on the number of web requests that AWS WAF evaluates using the rule group. You can help reduce these costs by using a scope-down statement to limit the requests that the rule group evaluates, such as limits by paths or content types. You might find that some parts of your application require more protection than others. For example, you may want to allow your homepage to load for everyone, including bots, but block requests to your application APIs.
- **Labels and label matching rules** – You can use the AWS WAF label match rule statement to evaluate the labels that the Bot Control rule group adds to your web requests. This allows you to customize how you handle web requests that are identified by the Bot Control managed rule group. For more information about labeling and using label match statements, see Label match rule statement (p. 66) and AWS WAF labels on web requests (p. 54).
- **Custom requests and responses** – You can add custom headers to requests that you allow and you can send custom responses for requests that you block by pairing label matching with the AWS WAF custom request and response features. For more information about customizing requests and responses, see Customizing web requests and responses in AWS WAF (p. 89).

Configuring and testing AWS WAF Bot Control

This section provides general guidance for configuring and testing an AWS WAF Bot Control implementation for your site. The specific steps that you choose to follow will depend on your needs, resources, and web requests that you receive.

> **Note**
> AWS Managed Rules are designed to protect you from common web threats. When used in accordance with the documentation, AWS Managed Rules rule groups add another layer of security for your applications. However, AWS Managed Rules rule groups aren't intended as a replacement for your security responsibilities, which are determined by the AWS resources that you select. Refer to the Shared Responsibility Model to ensure that your resources in AWS are properly protected.
>
> **Production traffic risk**
> Before you deploy your Bot Control implementation for production traffic, test and tune it in a staging or testing environment until you are comfortable with the potential impact to your traffic. Then test and tune the rules in count mode with your production traffic before enabling them.

This guidance is intended for users who know generally how to create and manage AWS WAF web ACLs, rules, and rule groups. Those topics are covered in prior sections of this guide.

To configure and test a Bot Control implementation

Perform these steps first in a test environment, then in production.

1. **Add the Bot Control managed rule group**

   Add the managed AWS rule group AWSManagedRulesBotControlRuleSet to a new or existing web ACL and configure it so that it doesn't alter current web ACL behavior.

   - When you add the managed rule group, edit it and, in the **Rules** pane, turn on the **Set all rule actions to count** toggle. With this configuration, AWS WAF evaluates requests against all of the rules in the rule group and only counts the matches that result, while still adding labels to requests. For more information, see Setting rule actions to count in a rule group (p. 24).

     This allows you to monitor the impact of the Bot Control rules to determine whether you want to add exceptions, such as exceptions for internal use cases or for desired bots.
   - Position the rule group so that it's evaluated last in the web ACL, with a priority setting that's numerically higher than any other rules or rule groups that you're already using. For more information, see To create a web ACL (p. 22).

     This way, your current handling of traffic isn't disrupted. For example, if you have rules that detect malicious traffic such as SQL injection or cross-site scripting, they'll continue to detect and log that. Alternately, if you have rules that allow known non-malicious traffic, they can continue to allow that traffic, without having it blocked by the Bot Control managed rule group. You might decide to adjust the processing order during your testing and tuning activities.
2. **Enable sampling, logging, and metrics for the web ACL**

   As needed, configure logging for the web ACL, and enable sampling and Amazon CloudWatch metrics. This allows you to monitor the interaction of the Bot Control managed rule group with your traffic.

   - For information about configuring and using logging, see Logging web ACL traffic information (p. 107).
   - For information about Amazon CloudWatch metrics, see Monitoring with Amazon CloudWatch (p. 370).
   - For information about web request sampling, see Viewing a sample of web requests (p. 28).
3. **Associate the web ACL with a resource**

   If the web ACL isn't already associated with a resource, associate it. For information, see Associating or disassociating a web ACL with an AWS resource (p. 26).
4. **Monitor traffic and Bot Control rule matches**

   Make sure that traffic is flowing and that the Bot Control managed rule group rules are adding labels to matching web requests. You can see the labels in the logs and see bot and label metrics in the Amazon CloudWatch metrics. In the logs, the rules that you've set to count in the rule group show up under `excludedRules` in the `ruleGroupList`.

   > **Note**
   > The Bot Control managed rule group verifies bots using the IP addresses from AWS WAF. If you use Bot Control and you have verified bots that route through a proxy or load balancer, you might need to explicitly allow them using a custom rule. For information about how to create a custom rule, see Forwarded IP address (p. 82). For information about how you can use the rule to customize Bot Control web request handling, see the next step.
5. **Customize Bot Control web request handling**

   As needed, add your own rules that explicitly allow or block requests, to change how Bot Control rules would otherwise handle them.

How you do this depends on your use case, but the following are common solutions:

- Explicitly allow requests with a rule that you add before the Bot Control managed rule group. With this, the allowed requests never reach the rule group for evaluation. This can help contain the cost of using the Bot Control managed rule group.
- Exclude requests from Bot Control evaluation with a scope-down statement inside the Bot Control managed rule group statement. This functions the same as the preceding option. It can help contain the cost of using the Bot Control managed rule group because the requests that don't match the scope-down statement never reach rule group evaluation. For information about scope-down statements, see Scope-down statements (p. 81).

  For examples, see AWS WAF Bot Control example: Exclude IP range from bot management (p. 105) and AWS WAF Bot Control example: Allow traffic from a bot that you control (p. 106).
- Use Bot Control labels in request handling to allow or block requests. Add a label match rule after the Bot Control managed rule group to filter out labeled requests that you want to allow from those that you want to block. After testing, keep the related Bot Control rules in count mode, and maintain the request handling decisions in your custom rule. For information about label match statements, see Label match rule statement (p. 66).

  For examples, see AWS WAF Bot Control example: Block verified bots (p. 99), AWS WAF Bot Control example: Allow a specific blocked bot (p. 100), and AWS WAF Bot Control example: Create an exception for a blocked user agent (p. 102).

For more examples, see AWS WAF Bot Control examples (p. 98).

6. **As needed, enable the Bot Control managed rule group settings**

   Depending on your situation, you might have decided that you want to leave some Bot Control rules in count mode. For the rules that you want to have run as they are configured inside the rule group, enable the regular rule configuration. To do this, disable count mode in the web ACL rule group configuration for the rules.
7. **Monitor and tune**

   To be sure that web requests are being handled as you want, closely monitor your traffic after you enable the Bot Control functionality that you intend to use. Adjust the behavior as needed with the rules count override on the rule group and with your own rules.

False positives with AWS WAF Bot Control

We have carefully selected the rules in the AWS WAF Bot Control managed rule group to minimize false positives. We test the rules against global traffic and monitor their impact on test web ACLs. However, it's still possible to get false positives occasionally.

Examples of situations where you might encounter false positives include the following:

- A rule that has a historically low false positive rate might have increased false positives for valid traffic. This might be due to new traffic patterns or request attributes that emerge with valid traffic, causing it to match the rule where it didn't before. These changes might be due to situations like the following:
  - Traffic details that are altered as traffic flows through network appliances, such as load balancers or content distribution networks (CDN).
  - Emerging changes in traffic data, for example new browsers or new versions for existing browsers.
- A rule with a low global false positive rate might heavily impact specific devices or applications. For example, in testing and validation, we might not have observed requests from applications with low traffic volumes or from less common browsers or devices.

- You might rely on some specific bot traffic for things like uptime monitoring, integration testing, or marketing tools. If Bot Control identifies and blocks the bot traffic that you want to allow, you need to alter the handling by adding your own rules. While this isn't a false positive scenario for all customers, if it is for you, the handling is the same as for a false positive.
- The Bot Control managed rule group verifies bots using the IP addresses from AWS WAF. If you use Bot Control and you have verified bots that route through a proxy or load balancer, you might need to explicitly allow them using a custom rule. For information about how to create a custom rule of this type, see Forwarded IP address (p. 82).

For information about how to handle false positives that you might get from the AWS WAF Bot Control managed rule group, see the guidance in the prior section, Configuring and testing AWS WAF Bot Control (p. 95).

AWS WAF Bot Control examples

This section shows example configurations that satisfy a variety of common use cases for AWS WAF Bot Control implementations.

Each example provides a description of the use case and then shows the solution in JSON listings for the custom configured rules.

> **Note**
> The JSON listings shown in these examples were created in the console by configuring the rule and then editing it using the **Rule JSON editor**. You can also retrieve the JSON for the rules in an entire rule group or web ACL using `get` commands through the APIs or the command line interface.

Topics

AWS WAF Bot Control example: Simple configuration

The following JSON listing shows an example web ACL with an AWS WAF Bot Control managed rule group. Note the visibility configuration, which allows you to get sampling and metrics for monitoring purposes.

```
{
  "Name": "Bot-Beta-WebACL",
  "Id": "...",
  "ARN": "...",
  "DefaultAction": {
    "Allow": {}
  },
  "Description": "Bot-Beta-WebACL",
  "Rules": [
```

```
    {
      ...
    },
    {
       "Name": "AWS-AWSBotControl-Example",
       "Priority": 5,
       "Statement": {
          "ManagedRuleGroupStatement": {
             "VendorName": "AWS",
             "Name": "AWSManagedRulesBotControlRuleSet"
          },
          "VisibilityConfig": {
             "SampledRequestsEnabled": true,
             "CloudWatchMetricsEnabled": true,
             "MetricName": "AWS-AWSBotControl-Example"
           }
       }
    ],
    "VisibilityConfig": {
      ...
    },
    "Capacity": 1496,
    "ManagedByFirewallManager": false
}
```

AWS WAF Bot Control example: Explicitly allow verified bots

AWS WAF Bot Control doesn't block bots that are known by AWS to be common and verifiable bots, like `googlebot`. When Bot Control identifies a web request as coming from a verified bot, it adds a label that names the bot and a label that indicates that it's a verified bot. Bot Control doesn't add any other labels, such as signals labels, in order to prevent known good bots from being blocked.

You might have other AWS WAF rules that block verified bots. If you want to ensure that verified bots are allowed, add a custom rule to allow them based on the Bot Control labels. Your new rule must run after the Bot Control managed rule group, so that the labels are available to match against.

The following rule explicitly allows verified bots.

```
{
  "Rule": {
    "Name": "match_rule",
    "Statement": {
      "LabelMatchStatement": {
        "Scope": "LABEL",
        "Key": "awswaf:managed:aws:bot-control:bot:verified"
      }
    },
    "RuleLabels": [],
    "Action": {
      "Allow": {}
    }
  }
}
```

AWS WAF Bot Control example: Block verified bots

In order to block verified bots, you must add a rule to block them that runs after the AWS WAF Bot Control managed rule group. To do this, identify the bot names that you want to block and use a label match statement to identify and block them. If you want to just block all verified bots, you can omit the match against the `bot:name:` label.

The following rule blocks only the `bingbot` verified bot. This rule must run after the Bot Control managed rule group.

```
{
  "Rule": {
    "Name": "match_rule",
    "Statement": {
      "AndStatement": {
        "Statements": [
          {
            "LabelMatchStatement": {
              "Scope": "LABEL",
              "Key": "awswaf:managed:aws:bot-control:bot:name:bingbot"
            }
          },
          {
            "LabelMatchStatement": {
              "Scope": "LABEL",
              "Key": "awswaf:managed:aws:bot-control:bot:verified"
            }
          }
        ]
      }
    },
    "RuleLabels": [],
    "Action": {
      "Block": {}
    }
  }
}
```

The following rule blocks all verified bots.

```
{
  "Rule": {
    "Name": "match_rule",
    "Statement": {
      "LabelMatchStatement": {
        "Scope": "LABEL",
        "Key": "awswaf:managed:aws:bot-control:bot:verified"
      }
    },
    "RuleLabels": [],
    "Action": {
      "Block": {}
    }
  }
}
```

AWS WAF Bot Control example: Allow a specific blocked bot

If AWS WAF Bot Control is blocking a bot that you do not want to block, do the following:

1. Identify the Bot Control rule that's blocking the bot by checking the logs. For information about the web ACL logs, see Logging web ACL traffic information (p. 107). Note any identifying features like labels and user agent.
2. In your web ACL, exclude the rule that's blocking the bot from the rule group. To do this in the console, you edit the rule group inside the web ACL and set the blocking rule to count. This ensures that the bot is not blocked, while still allowing the rule to apply its label to matching requests.

At this point, you've configured the web ACL to stop blocking all bots that are normally blocked by the Bot Control rule.

3. Add a custom label matching rule to your web ACL, after the Bot Control managed rule group. Configure the rule to match against the Bot Control rule label and to block all matching requests except for the bot that you don't want to block.

   Your web ACL is now configured so that the bot you wanted to allow is no longer blocked by the Bot Control managed rule group.

For example, suppose you want to block all monitoring bots except for `pingdom`. In this case, you exclude the `CategoryMonitoring` rule and then write a rule to block all monitoring bots except for those with the bot name label `pingdom`.

The following rule uses the Bot Control managed rule group but changes the rule action for `CategoryMonitoring` to count, by excluding it from normal rule group processing. The category monitoring rule applies its labels as usual to matching requests, but only counts them instead of performing its usual action of block.

```
{
  "Name": "AWS-AWSBotControl-Example",
  "Priority": 5,
  "Statement": {
    "ManagedRuleGroupStatement": {
      "VendorName": "AWS",
      "Name": "AWSManagedRulesBotControlRuleSet",
      "ExcludedRules": [
        {
          "Name": "CategoryMonitoring"
        }
      ]
    }
  },
  "VisibilityConfig": {
    "SampledRequestsEnabled": true,
    "CloudWatchMetricsEnabled": true,
    "MetricName": "AWS-AWSBotControl-Example"
  }
}
```

The following rule matches against the category monitoring label that the preceding `CategoryMonitoring` rule adds to matching web requests. Among the category monitoring requests, this rule blocks all but those that have a label for the bot name `pingdom`.

The following rule must run after the preceding Bot Control managed rule group in the web ACL processing order.

```
{
  "Rule": {
    "Name": "match_rule",
    "Statement": {
      "AndStatement": {
        "Statements": [
          {
            "LabelMatchStatement": {
              "Scope": "LABEL",
              "Key": "awswaf:managed:aws:bot-control:bot:category:monitoring"
            }
          },
          {
```

```
            "NotStatement": {
              "Statement": {
                "LabelMatchStatement": {
                  "Scope": "LABEL",
                  "Key": "awswaf:managed:aws:bot-control:bot:name:pingdom"
                }
              }
            }
          }
        ]
      }
    },
    "RuleLabels": [],
    "Action": {
      "Block": {}
    }
  }
}
```

AWS WAF Bot Control example: Create an exception for a blocked user agent

If a bot is being erroneously blocked, you can create an exception by excluding the offending AWS WAF Bot Control rule and then combining the rule label with the exception criteria.

The following rule uses the Bot Control managed rule group but changes the rule action for `SignalNonBrowserUserAgent` to count, by excluding it from normal rule group processing. The signal rule applies its labels as usual to matching requests, but only counts them instead of performing its usual action of block.

```
{
  "Name": "AWS-AWSBotControl-Example",
  "Priority": 5,
  "Statement": {
    "ManagedRuleGroupStatement": {
      "VendorName": "AWS",
      "Name": "AWSManagedRulesBotControlRuleSet",
      "ExcludedRules": [
        {
          "Name": "SignalNonBrowserUserAgent"
        }
      ]
    }
  },
  "VisibilityConfig": {
    "SampledRequestsEnabled": true,
    "CloudWatchMetricsEnabled": true,
    "MetricName": "AWS-AWSBotControl-Example"
  }
}
```

The following rule matches against the signal label that the preceding Bot Control excluded rule adds to its matching web requests. Among the signal requests, this rule blocks all but those that have the user agent that you want to allow.

The following rule must run after the preceding Bot Control managed rule group in the web ACL processing order.

```
{
  "Rule": {
```

```
    "Name": "match_rule",
    "Statement": {
      "AndStatement": {
        "Statements": [
          {
            "LabelMatchStatement": {
              "Scope": "LABEL",
              "Key": "awswaf:managed:aws:bot-control:signal:non_browser_user_agent"
            }
          },
          {
            "NotStatement": {
              "Statement": {
                "ByteMatchStatement": {
                  "FieldToMatch": {
                    "SingleHeader": {
                      "Name": "user-agent"
                    }
                  },
                  "PositionalConstraint": "EXACTLY",
                  "SearchString": "TW96aWxsYS81LjAgQSBGcmllbmRseSBPbmU=",
                  "TextTransformations": [
                    {
                      "Priority": 0,
                      "Type": "NONE"
                    }
                  ]
                }
              }
            }
          }
        ]
      }
    },
    "RuleLabels": [],
    "Action": {
      "Block": {}
    },
    "VisibilityConfig": {
      "SampledRequestsEnabled": true,
      "CloudWatchMetricsEnabled": true,
      "MetricName": "match_rule"
    }
  }
}
```

AWS WAF Bot Control example: Use Bot Control only for login page

The following example uses a scope-down statement to use AWS WAF Bot Control only for traffic that's coming to a website's login page that's identified by the URI path `login`. The URI path to your login page might be different from the example, depending on your application and environment.

```
{
  "Name": "AWS-AWSBotControl-Example",
  "Priority": 5,
  "Statement": {
    "ManagedRuleGroupStatement": {
      "VendorName": "AWS",
      "Name": "AWSManagedRulesBotControlRuleSet",
      "ExcludedRules": [
        {
          "Name": "CategoryVerifiedSearchEngine"
```

```
        },
        {
          "Name": "CategoryVerifiedSocialMedia"
        }
      ]
    },
    "VisibilityConfig": {
      "SampledRequestsEnabled": true,
      "CloudWatchMetricsEnabled": true,
      "MetricName": "AWS-AWSBotControl-Example"
    },
    "ScopeDownStatement": {
      "ByteMatchStatement": {
        "SearchString": "login",
        "FieldToMatch": {
          "UriPath": {}
        },
        "TextTransformations": [
          {
            "Priority": 0,
            "Type": "NONE"
          }
        ],
        "PositionalConstraint": "CONTAINS"
      }
    }
  }
}
```

AWS WAF Bot Control example: Use Bot Control only for dynamic content

This example uses a scope-down statement to apply AWS WAF Bot Control only to dynamic content.

The scope-down statement excludes static content by negating the match results for a regex pattern set:

- The regex pattern set is configured to match extensions of *static content*. For example, the regex pattern set specification might be `(?i)\.(jpe?g|gif|png|svg|ico|css|js|woff2?)$`. For information about regex pattern sets and statements, see Regex pattern set match rule statement (p. 70).
- In the scope-down statement, we exclude the matching static content by nesting the regex pattern set statement inside a `NOT` statement. For information about the `NOT` statement, see `NOT` rule statement (p. 67).

```
{
  "Name": "AWS-AWSBotControl-Example",
  "Priority": 5,
  "Statement": {
    "ManagedRuleGroupStatement": {
      "VendorName": "AWS",
      "Name": "AWSManagedRulesBotControlRuleSet",
      "ExcludedRules": [
        {
          "Name": "CategoryVerifiedSearchEngine"
        },
        {
          "Name": "CategoryVerifiedSocialMedia"
        }
      ]
    },
```

```
    "VisibilityConfig": {
      "SampledRequestsEnabled": true,
      "CloudWatchMetricsEnabled": true,
      "MetricName": "AWS-AWSBotControl-Example"
    },
    "ScopeDownStatement": {
      "NotStatement": {
        "Statement": {
          "RegexPatternSetReferenceStatement": {
            "ARN": "arn:aws:wafv2:us-east-1:123456789:regional/regexpatternset/
excludeset/00000000-0000-0000-0000-000000000000",
            "FieldToMatch": {
              "UriPath": {}
            },
            "TextTransformations": [
              {
                "Priority": 0,
                "Type": "NONE"
              }
            ]
          }
        }
      }
    }
  }
}
```

AWS WAF Bot Control example: Exclude IP range from bot management

If you want to exclude a subset of web traffic from AWS WAF Bot Control management, and you can identify that subset using a rule statement, then exclude it by adding a scope-down statement to your Bot Control managed rule group statement.

The following rule performs normal Bot Control bot management on all web traffic except for web requests coming from a specific IP address range.

```
{
  "Name": "AWS-AWSBotControl-Example",
  "Priority": 5,
  "Statement": {
    "ManagedRuleGroupStatement": {
      "VendorName": "AWS",
      "Name": "AWSManagedRulesBotControlRuleSet",
      "ExcludedRules": [
        {
          "Name": "CategoryVerifiedSearchEngine"
        },
        {
          "Name": "CategoryVerifiedSocialMedia"
        }
      ]
    },
    "VisibilityConfig": {
      "SampledRequestsEnabled": true,
      "CloudWatchMetricsEnabled": true,
      "MetricName": "AWS-AWSBotControl-Example"
    },
    "ScopeDownStatement": {
      "NotStatement": {
        "Statement": {
          "IPSetReferenceStatement": {
```

```
            "ARN": "arn:aws:wafv2:us-east-1:123456789:regional/ipset/
friendlyips/00000000-0000-0000-0000-000000000000"
          }
        }
      }
    }
  }
}
```

AWS WAF Bot Control example: Allow traffic from a bot that you control

You can configure some site monitoring bots and custom bots to send custom headers. If you want to allow traffic from a bot like this, you can configure it to add a shared secret in a header. Then you exclude messages that have the header by adding a scope-down statement to the AWS WAF Bot Control managed rule group statement.

The following example rule excludes traffic with a secret header from Bot Control inspection.

```
{
  "Name": "AWS-AWSBotControl-Example",
  "Priority": 5,
  "Statement": {
    "ManagedRuleGroupStatement": {
      "VendorName": "AWS",
      "Name": "AWSManagedRulesBotControlRuleSet",
      "ExcludedRules": [
        {
          "Name": "CategoryVerifiedSearchEngine"
        },
        {
          "Name": "CategoryVerifiedSocialMedia"
        }
      ]
    },
    "VisibilityConfig": {
      "SampledRequestsEnabled": true,
      "CloudWatchMetricsEnabled": true,
      "MetricName": "AWS-AWSBotControl-Example"
    },
    "ScopeDownStatement": {
      "NotStatement": {
        "Statement": {
          "ByteMatchStatement": {
            "SearchString": "YSBzZWNyZXQ=",
            "FieldToMatch": {
              "SingleHeader": {
                "Name": "x-bypass-secret"
              }
            },
            "TextTransformations": [
              {
                "Priority": 0,
                "Type": "NONE"
              }
            ],
            "PositionalConstraint": "EXACTLY"
          }
        }
      }
    }
  }
```

```
}
```

Logging web ACL traffic information

You can enable logging to get detailed information about traffic that is analyzed by your web ACL. Information that is contained in the logs includes the time that AWS WAF received the request from your AWS resource, detailed information about the request, and the action for the rule that each request matched.

In the logging configuration for your web ACL, you can customize what AWS WAF sends to the logs as follows:

- **Log filtering** – You can add filtering to specify which web requests are kept in the logs and which are dropped. You can filter on the rule action and on the web request labels that were applied during the request evaluation. For information about rule action settings, see AWS WAF rule action (p. 53). For information about labels, see AWS WAF labels on web requests (p. 54).
- **Field redaction** – You can redact some fields from the log records. Redacted fields appear as `XXX` in the logs. For example, if you redact the **URI** field, the **URI** field in the logs will be `XXX`. For a list of the log fields, see Log Fields (p. 108).

You send logs from your web ACL to an Amazon Kinesis Data Firehose with a configured storage destination. After you enable logging, AWS WAF delivers logs to your storage destination through the HTTPS endpoint of Kinesis Data Firehose.

For information about how to create an Amazon Kinesis Data Firehose and review your stored logs, see What Is Amazon Kinesis Data Firehose? To understand the permissions required for your Kinesis Data Firehose configuration, see Controlling Access with Amazon Kinesis Data Firehose.

You must have the following permissions to successfully enable logging:

- `iam:CreateServiceLinkedRole`
- `firehose:ListDeliveryStreams`
- `wafv2:PutLoggingConfiguration`

For more information about service-linked roles and the `iam:CreateServiceLinkedRole` permission, see Using service-linked roles for AWS WAF (p. 130).

Managing logging for a web ACL

You can enable and disable logging for a web ACL at any time.

To enable logging for a web ACL

1. Create an Amazon Kinesis Data Firehose using a name starting with the prefix `aws-waf-logs-`. For example, `aws-waf-logs-us-east-2-analytics`. Create the data firehose with a `PUT` source and in the region that you are operating. If you are capturing logs for Amazon CloudFront, create the firehose in US East (N. Virginia). For more information, see Creating an Amazon Kinesis Data Firehose Delivery Stream.

   Important
   Do not choose `Kinesis stream` as your source.
   One AWS WAF log is equivalent to one Kinesis Data Firehose record. If you typically receive 10,000 requests per second and you enable full logs, you should have a 10,000 records per second setting in Kinesis Data Firehose. If you don't configure Kinesis Data Firehose

correctly, AWS WAF won't record all logs. For more information, see Amazon Kinesis Data Firehose Quotas.

2. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
3. In the navigation pane, choose **Web ACLs**.
4. Choose the web ACL that you want to enable logging for.
5. On the **Logging** tab, choose **Enable logging**.
6. Choose the Kinesis Data Firehose that you created in the first step. You must choose a firehose that begins with `aws-waf-logs-`.
7. (Optional) If you don't want certain fields and their values included in the logs, redact those fields. Choose the field to redact, and then choose **Add**. Repeat as necessary to redact additional fields. The redacted fields appear as `XXX` in the logs. For example, if you redact the **URI** field, the **URI** field in the logs will be `XXX`.
8. (Optional) If you don't want to send all requests to the logs, add your filtering criteria and behavior. Under **Filter logs**, for each filter that you want to apply, choose **Add filter**, then choose your filtering criteria and specify whether you want to keep or drop requests that match the criteria. When you finish adding filters, if needed, modify the **Default logging behavior**.
9. Choose **Enable logging**.

   Note
   When you successfully enable logging, AWS WAF will create a service linked role with the necessary permissions to write logs to the Amazon Kinesis Data Firehose. For more information, see Using service-linked roles for AWS WAF (p. 130).

To disable logging for a web ACL

1. In the navigation pane, choose **Web ACLs**.
2. Choose the web ACL that you want to disable logging for.
3. On the **Logging** tab, choose **Disable logging**.
4. In the dialog box, choose **Disable logging**.

Log Fields

The following list describes the possible log fields.

action

The action. Possible values for a terminating rule: `ALLOW` and `BLOCK`. `COUNT` is a non terminating rule action.

args

The query string.

clientIp

The IP address of the client sending the request.

country

The source country of the request. If AWS WAF is unable to determine the country of origin, it sets this field to `-`.

excludedRules

The list of rules in the rule group that you have excluded. The action for these rules is set to COUNT.

exclusionType

A type that indicates that the excluded rule has the action COUNT.

ruleId

The ID of the rule within the rule group that is excluded.

formatVersion

The format version for the log.

headers

The list of headers.

httpMethod

The HTTP method in the request.

httpRequest

The metadata about the request.

httpSourceId

The source ID. This field shows the ID of the associated resource.

httpSourceName

The source of the request. Possible values: `CF` for Amazon CloudFront, `APIGW` for Amazon API Gateway, `ALB` for Application Load Balancer, and `APPSYNC` for AWS AppSync.

httpVersion

The HTTP version.

labels

The labels on the web request. These labels were applied by rules that were used to evaluate the request.

limitKey

Indicates the IP address source that AWS WAF should use to aggregate requests for rate limiting by a rate-based rule. Possible values are `IP`, for web request origin, and `FORWARDED_IP`, for an IP forwarded in a header in the request.

limitValue

The IP address used by a rate-based rule to aggregate requests for rate limiting. If a request contains an IP address that isn't valid, the `limitvalue` is `INVALID`.

maxRateAllowed

The maximum number of requests, which have an identical value in the field that is specified by `limitKey`, allowed in a five-minute period. If the number of requests exceeds the `maxRateAllowed` and the other predicates specified in the rule are also met, AWS WAF triggers the action that is specified for this rule.

nonTerminatingMatchingRules

The list of non-terminating rules in the rule group that match the request. These are always COUNT rules (non-terminating rules that match).

action

This is always COUNT (non-terminating rules that match).

ruleId

The ID of the rule within the rule group that matches the request and was non-terminating. That is, COUNT rules.

ruleMatchDetails

Detailed information about the rule that matched the request. This field is only populated for SQL injection and cross-site scripting (XSS) match rule statements.

rateBasedRuleId

The ID of the rate-based rule that acted on the request. If this has terminated the request, the ID for `rateBasedRuleId` is the same as the ID for `terminatingRuleId`.

rateBasedRuleList

The list of rate-based rules that acted on the request.

requestHeadersInserted

The list of headers inserted for custom request handling.

requestId

The ID of the request, which is generated by the underlying host service. For Application Load Balancer, this is the trace ID. For all others, this is the request ID.

responseCodeSent

The response code sent with a custom response.

ruleGroupId

The ID of the rule group. If the rule blocked the request, the ID for `ruleGroupID` is the same as the ID for `terminatingRuleId`.

ruleGroupList

The list of rule groups that acted on this request.

terminatingRule

The rule within the rule group that terminated the request. If this is a non-null value, it also contains a **ruleId** and **action**.

terminatingRuleId

The ID of the rule that terminated the request. If nothing terminates the request, the value is `Default_Action`.

terminatingRuleMatchDetails

Detailed information about the terminating rule that matched the request. A terminating rule has an action that ends the inspection process against a web request. Possible actions for a terminating rule are `ALLOW` and `BLOCK`. This is only populated for SQL injection and cross-site scripting (XSS) match rule statements. As with all rule statements that inspect for more than one thing, AWS WAF applies the action on the first match and stops inspecting the web request. A web request with a terminating action could contain other threats, in addition to the one reported in the log.

terminatingRuleType

The type of rule that terminated the request. Possible values: RATE_BASED, REGULAR, GROUP, and MANAGED_RULE_GROUP.

timestamp

The timestamp in milliseconds.

uri

The URI of the request. The preceding code example demonstrates what the value would be if this field had been redacted.

webaclId

The GUID of the web ACL.

Log Examples

Example Log output for a rule that triggered on SQLi detection (terminating)


```
{
    "timestamp": 1576280412771,
    "formatVersion": 1,
    "webaclId": "arn:aws:wafv2:ap-southeast-2:EXAMPLE12345:regional/webacl/
STMTest/1EXAMPLE-2ARN-3ARN-4ARN-123456EXAMPLE",
    "terminatingRuleId": "STMTest_SQLi_XSS",
    "terminatingRuleType": "REGULAR",
    "action": "BLOCK",
    "terminatingRuleMatchDetails": [
        {
            "conditionType": "SQL_INJECTION",
            "location": "HEADER",
            "matchedData": [
                "10",
                "AND",
                "1"
            ]
        }
    ],
    "httpSourceName": "-",
    "httpSourceId": "-",
    "ruleGroupList": [],
    "rateBasedRuleList": [],
    "nonTerminatingMatchingRules": [],
    "httpRequest": {
        "clientIp": "1.1.1.1",
        "country": "AU",
        "headers": [
            {
                "name": "Host",
                "value": "localhost:1989"
            },
            {
                "name": "User-Agent",
                "value": "curl/7.61.1"
            },
            {
                "name": "Accept",
                "value": "*/*"
            },
            {
                "name": "x-stm-test",
                "value": "10 AND 1=1"
            }
        ],
        "uri": "/foo",
        "args": "",
        "httpVersion": "HTTP/1.1",
        "httpMethod": "GET",
        "requestId": "rid"
    },
    "labels": [
        {
            "name": "value"
        }
```

```
    ]
}
```

Example Log output for a rule that triggered on SQLi detection (non-terminating)

```
{
    "timestamp":1592357192516
    ,"formatVersion":1
    ,"webaclId":"arn:aws:wafv2:us-east-1:123456789012:global/webacl/hello-
world/5933d6d9-9dde-js82-v8aw-9ck28nv9"
    ,"terminatingRuleId":"Default_Action"
    ,"terminatingRuleType":"REGULAR"
    ,"action":"ALLOW"
    ,"terminatingRuleMatchDetails":[]
    ,"httpSourceName":"-"
    ,"httpSourceId":"-"
    ,"ruleGroupList":[]
    ,"rateBasedRuleList":[]
    ,"nonTerminatingMatchingRules":
    [{
        "ruleId":"TestRule"
        ,"action":"COUNT"
        ,"ruleMatchDetails":
        [{
            "conditionType":"SQL_INJECTION"
            ,"location":"HEADER"
            ,"matchedData":[
                "10"
                ,"and"
                ,"1"]
            }]
    }]
    ,"httpRequest":{
        "clientIp":"3.3.3.3"
        ,"country":"US"
        ,"headers":[
            {"name":"Host","value":"localhost:1989"}
            ,{"name":"User-Agent","value":"curl/7.61.1"}
            ,{"name":"Accept","value":"*/*"}
            ,{"name":"foo","value":"10 AND 1=1"}
            ]
            ,"uri":"/foo","args":""
            ,"httpVersion":"HTTP/1.1"
            ,"httpMethod":"GET"
            ,"requestId":"rid"
    },
    "labels": [
        {
            "name": "value"
        }
    ]
}
```

Example Log output for multiple rules that triggered inside a rule group (RuleA-XSS is terminating and Rule-B is non-terminating)

```
{
    "timestamp":1592361810888,
    "formatVersion":1,
    "webaclId":"arn:aws:wafv2:us-east-1:123456789012:global/webacl/hello-
world/5933d6d9-9dde-js82-v8aw-9ck28nv9"
    ,"terminatingRuleId":"RG-Reference"
    ,"terminatingRuleType":"GROUP"
```

```
    ,"action":"BLOCK",
    "terminatingRuleMatchDetails":
    [{
        "conditionType":"XSS"
        ,"location":"HEADER"
        ,"matchedData":["<","frameset"]
    }]
    ,"httpSourceName":"-"
    ,"httpSourceId":"-"
    ,"ruleGroupList":
    [{
        "ruleGroupId":"arn:aws:wafv2:us-east-1:123456789012:global/rulegroup/hello-world/
c05lb698-1f11-4m41-aef4-99a506d53f4b"
        ,"terminatingRule":{
            "ruleId":"RuleA-XSS"
            ,"action":"BLOCK"
            ,"ruleMatchDetails":null
            }
        ,"nonTerminatingMatchingRules":
        [{
            "ruleId":"RuleB-SQLi"
            ,"action":"COUNT"
            ,"ruleMatchDetails":
            [{
                "conditionType":"SQL_INJECTION"
                ,"location":"HEADER"
                ,"matchedData":[
                    "10"
                    ,"and"
                    ,"1"]
            }]
        }]
        ,"excludedRules":null
    }]
    ,"rateBasedRuleList":[]
    ,"nonTerminatingMatchingRules":[]
    ,"httpRequest":{
        "clientIp":"3.3.3.3"
        ,"country":"US"
        ,"headers":
        [
            {"name":"Host","value":"localhost:1989"}
            ,{"name":"User-Agent","value":"curl/7.61.1"}
            ,{"name":"Accept","value":"*/*"}
            ,{"name":"xssfoo","value":"<frameset onload=alert(1)>"}
            ,{"name":"bar","value":"10 AND 1=1"}
            ]
        ,"uri":"/foo"
        ,"args":""
        ,"httpVersion":"HTTP/1.1"
        ,"httpMethod":"GET"
        ,"requestId":"rid"
    },
    "labels": [
        {
            "name": "value"
        }
    ]
}
```


Example Log output for a rule that triggered for the inspection of the request body with content type JSON

AWS WAF currently reports the location for JSON body inspection as `UNKNOWN`.

```
{
    "timestamp": 1576280412771,
    "formatVersion": 1,
    "webaclId": "arn:aws:wafv2:ap-southeast-2:12345:regional/webacl/test/111",
    "terminatingRuleId": "STMTest_SQLi_XSS",
    "terminatingRuleType": "REGULAR",
    "action": "BLOCK",
    "terminatingRuleMatchDetails": [
        {
            "conditionType": "SQL_INJECTION",
            "location": "UNKNOWN",
            "matchedData": [
                "10",
                "AND",
                "1"
            ]
        }
    ],
    "httpSourceName": "ALB",
    "httpSourceId": "alb",
    "ruleGroupList": [],
    "rateBasedRuleList": [],
    "nonTerminatingMatchingRules": [],
    "requestHeadersInserted":null,
    "responseCodeSent":null,
    "httpRequest": {
        "clientIp": "1.1.1.1",
        "country": "AU",
        "headers": [],
        "uri": "",
        "args": "",
        "httpVersion": "HTTP/1.1",
        "httpMethod": "POST",
        "requestId": "null"
    },
    "labels": [
        {
            "name": "value"
        }
    ]
}
```


Listing IP addresses blocked by rate-based rules

You can access the list of IP addresses that are currently blocked by a rate-based rule by using the CLI, the API, or any of the SDKs. This topic covers access using the CLI and APIs. The console doesn't provide this functionality at this time.

For the AWS WAF API, the command is GetRateBasedStatementManagedKeys.

For the AWS WAF CLI, the command is get-rate-based-statement-managed-keys.

The maximum number of IP addresses that can be blocked for a single rate-based rule instance is 10,000. If more than 10,000 addresses exceed the rate limit, AWS WAF blocks those with the highest rates.

The following shows the syntax for retrieving the list of blocked IP addresses for a rate-based rule that's being used in a web ACL on an Amazon CloudFront distribution.

```
aws wafv2 get-rate-based-statement-managed-keys --scope=CLOUDFRONT --region=us-east-1 --
web-acl-name=WebACLName --web-acl-id=WebACLId --rule-name=RuleName
```

The following shows the syntax for a regional application, an Amazon API Gateway REST API, an Application Load Balancer, or an AWS AppSync GraphQL API.

```
aws wafv2 get-rate-based-statement-managed-keys --scope=REGIONAL --region=region --web-acl-
name=WebACLName --web-acl-id=WebACLId --rule-name=RuleName
```

AWS WAF monitors web requests and manages keys independently for each unique combination of web ACL, optional rule group, and rate-based rule. For example, if you define a rate-based rule inside a rule group, and then use the rule group in a web ACL, AWS WAF monitors web requests and manages keys for that web ACL, rule group reference statement, and rate-based rule instance. If you use the same rule group in a second web ACL, AWS WAF monitors web requests and manages keys for this second usage completely independent of your first.

For a rate-based rule that you've defined inside a rule group, you need to provide the name of the rule group reference statement in your request, in addition to the web ACL name and the name of the rate-based rule name inside the rule group. The following shows the syntax for a regional application where the rate-based rule is defined inside a rule group, and the rule group is used in a web ACL.

```
aws wafv2 get-rate-based-statement-managed-keys --scope=REGIONAL --region=region --web-
acl-name=WebACLName --web-acl-id=WebACLId --rule-group-rule-name=RuleGroupRuleName --rule-
name=RuleName
```

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 deny list 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 web requests (p. 28).

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

Using AWS WAF with CloudFront for applications running on your own HTTP server

When you use AWS WAF with CloudFront, you can protect your applications running on any HTTP webserver, whether it's a webserver that's running in Amazon Elastic Compute Cloud (Amazon EC2) or a webserver that you manage privately. You can also configure CloudFront to require HTTPS between CloudFront and your own webserver, as well as between viewers and CloudFront.

Requiring HTTPS Between CloudFront and Your Own Webserver

To require HTTPS between CloudFront and your own webserver, you can use the CloudFront custom origin feature and configure the **Origin Protocol Policy** and the **Origin Domain Name** settings for specific origins. In your CloudFront configuration, you can specify the DNS name of the server along with the port and the protocol that you want CloudFront to use when fetching objects from your origin. You should also ensure that the SSL/TLS certificate on your custom origin server matches the origin domain name you've configured. When you use your own HTTP webserver outside of AWS, you must use a certificate that is signed by a trusted third-party certificate authority (CA), for example, Comodo, DigiCert, or Symantec. For more information about requiring HTTPS for communication between CloudFront and your own webserver, see the topic Requiring HTTPS for Communication Between CloudFront and Your Custom Origin in the *Amazon CloudFront Developer Guide*.

Requiring HTTPS Between a Viewer and CloudFront

To require HTTPS between viewers and CloudFront, you can change the **Viewer Protocol Policy** for one or more cache behaviors in your CloudFront distribution. For more information about using HTTPS between viewers and CloudFront, see the topic Requiring HTTPS for Communication Between Viewers and CloudFront in the *Amazon CloudFront Developer Guide*. You can also bring your own SSL certificate so viewers can connect to your CloudFront distribution over HTTPS using your own domain name, for example *https://www.mysite.com*. For more information, see the topic Configuring Alternate Domain Names and HTTPS 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 byte match rule statements to allow or block requests based on the HTTP method, as described in String match rule statement (p. 73). 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*.

Security in AWS WAF

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. The effectiveness of our security is regularly tested and verified by third-party auditors as part of the AWS compliance programs. To learn about the compliance programs that apply to AWS WAF, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your organization's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using AWS WAF. The following topics show you how to configure AWS WAF to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your AWS WAF resources.

Topics

- Resilience in AWS WAF (p. 135)
- Infrastructure security in AWS WAF (p. 135)

Data protection in AWS WAF

The AWS shared responsibility model applies to data protection in AWS WAF. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. This content includes the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form fields such as a **Name** field. This includes when you work with AWS WAF or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

AWS WAF entities—such as web ACLs, rule groups, and IP sets—are encrypted at rest, except in certain Regions where encryption is not available, including China (Beijing) and China (Ningxia). Unique encryption keys are used for each Region.

Deleting AWS WAF resources

You can delete the resources that you create in AWS WAF. See the guidance for each resource type in following sections.

- Deleting a web ACL (p. 26)
- Deleting a rule group (p. 52)
- Deleting an IP set (p. 86)
- Deleting a regex pattern set (p. 88)

Identity and access management in 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 first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.
- **IAM user** – An IAM user is 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 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 an IAM identity that you can create in your account that has specific permissions. An IAM role is similar to an IAM user in that it is an AWS identity with permissions policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session. IAM roles with temporary credentials are useful in the following situations:

  - **Federated user access** – Instead of creating an IAM user, you can use existing 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*.

  - **AWS service access** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. 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 that are running on an EC2 instance and making AWS CLI or 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 that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances 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 group*.

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. 120)
- Using identity-based policies (IAM policies) for AWS WAF (p. 124)
- AWS WAF API permissions: Actions, resources, and conditions reference (p. 128)

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). 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*, *rule groups*, *IP sets*, and *regex pattern sets*. 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 resources have the following format:

```
arn:aws:wafv2:region:account:scope/resource/resource/ID
```

The following table lists the format for each resource.

Name in AWS WAF Console	Name in AWS WAF SDK/CLI	ARN Format	
Web ACL	`WebACL`	`arn:aws:wafv2:region:account:scope/webacl/name/ID`	
Rule group	`RuleGroup`	`arn:aws:wafv2:region:account:scope/rulegroup/name/ID`	
IP set	`IPSet`	`arn:aws:wafv2:region:account:scope/ipset/name/ID`	
Regex pattern set	`RegexPatternSet`	`arn:aws:wafv2:region:account:scope/regexpatternset/name/ID`	

To specify an AWS WAF resource ARN, replace the variables in the ARN formats with valid values as follows:

- *region*: The AWS Region you're using. For Amazon CloudFront, set this to `us-east-1`. For regional resources, set this to the Region you're interested in.
- *account*: The ID of your AWS account.
- *scope*: The scope of the resource, which can be either `regional`, for use with an Amazon API Gateway REST API, an Application Load Balancer, or an AWS AppSync GraphQL API or `global`, for use with an Amazon CloudFront distribution.
- *name*: The name that you gave the AWS WAF resource, or a wildcard (`*`) to indicate all resources of the specified type that are associated with the specified AWS account. If you use the wildcard for the name, you must also use it for the ID.
- *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. If you use the wildcard for the ID, you must also use it for the name.

For example, the following ARN specifies all web ACLs with regional scope for the account `111122223333` in Region `us-east-1`:

```
arn:aws:wafv2:us-east-1:111122223333:regional/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 sections explain the available options for creating permissions policies.

> **Note**
> These sections discuss 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 Identity and Access Management 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 `wafv2:ListWebACLs` 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": "2019-07-29",
    "Statement": [
        {
            "Sid": "ListWebACLs",
            "Effect": "Allow",
            "Action": [
                "wafv2:ListWebACLs"
            ],
            "Resource": "*"
        }
    ]
}
```

For more information about using identity-based policies with AWS WAF, see Using identity-based policies (IAM policies) for AWS WAF (p. 124). 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. 121)), the service defines a set of API operations (see AWS WAF API permissions: Actions, resources, and conditions reference (p. 128)). 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. 121).
- **Action** – You use action keywords to identify resource operations that you want to allow or deny. For example, the `wafv2:CreateRuleGroup` permission allows the user permissions to perform the AWS WAF `CreateRuleGroup` 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 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 Identity and Access Management 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. 128).

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 general AWS condition keys that you can use as appropriate. For a complete list of AWS keys, see Available Keys for Conditions in the *IAM User Guide*.

Using identity-based policies (IAM policies) for AWS WAF

This section 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. 120).

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. 128).

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. 128). For more information about these additional console permissions, see Customer managed policy examples (p. 125).

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:

- `AWSWAFReadOnlyAccess` – Grants read-only access to AWS WAF resources.
- `AWSWAFFullAccess` – Grants full access to AWS WAF resources.
- `AWSWAFConsoleReadOnlyAccess` – Grants read-only access to the AWS WAF console, which includes resources for AWS WAF and integrated services, such as Amazon CloudFront, Amazon API Gateway, Application Load Balancer, and AWS AppSync.
- `AWSWAFConsoleFullAccess` – Grants full access to the AWS WAF console, which includes resources for AWS WAF and integrated services, such as Amazon CloudFront, Amazon API Gateway, Application Load Balancer, and AWS AppSync.

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 on the console. As you attach policies to the user, you can verify that the user can perform various operations on 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

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": [
            "wafv2:Get*",
            "wafv2:List*",
            "cloudfront:GetDistribution",
            "cloudfront:GetDistributionConfig",
            "cloudfront:ListDistributions",
            "cloudfront:ListDistributionsByWebACLId",
            "cloudwatch:ListMetrics",
            "cloudwatch:GetMetricStatistics",
            "ec2:DescribeRegions"
         ],
         "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": [
            "wafv2:*",
            "cloudfront:CreateDistribution",
            "cloudfront:GetDistribution",
            "cloudfront:GetDistributionConfig",
            "cloudfront:UpdateDistribution",
            "cloudfront:ListDistributions",
            "cloudfront:ListDistributionsByWebACLId",
            "cloudfront:DeleteDistribution",
            "cloudwatch:ListMetrics",
            "cloudwatch:GetMetricStatistics",
            "ec2:DescribeRegions"
         ],
         "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": [
            "wafv2:*"
         ],
         "Resource": [
            "arn:aws:wafv2:us-east-1:444455556666:*"
         ]
      },
      {
         "Effect": "Allow",
         "Action": [
            "cloudfront:GetDistribution",
            "cloudfront:GetDistributionConfig",
            "cloudfront:ListDistributions",
            "cloudfront:ListDistributionsByWebACLId",
            "cloudfront:UpdateDistribution",
            "cloudwatch:ListMetrics",
            "cloudwatch:GetMetricStatistics",
            "ec2:DescribeRegions"
         ],
         "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": [
            "wafv2:*"
         ],
         "Resource": [
        "arn:aws:wafv2:us-east-1:444455556666:regional/webacl/test123/112233d7c-86b2-458b-
af83-51c51example"
         ]
      }
   ]
}
```

AWS WAF API permissions: Actions, resources, and conditions reference

When you set up Access control (p. 120) and writing permissions policies that you can attach to an IAM identity (identity-based policies), use the information in this section as a guide. For each AWS WAF API operation, you need to know the 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.

> **Note**
> To specify an action, use the `wafv2:` prefix followed by the API operation name (for example, `wafv2:CreateIPSet`).

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

Global and regional settings

In the resource settings in this section, use the following `scope` and `region` settings:

- For CloudFront distributions, set *`scope`* to `global` and set *`region`* to `us-east-1`.
- For an Amazon API Gateway REST API, an Application Load Balancer, or an AWS AppSync GraphQL API, set *`scope`* to `regional` and set the *`region`* to the region you're interested in.

AWS WAF API permissions for references to resources

For all resource permissions settings, if a resource references any other resource by Amazon resource name (ARN), you must have permissions to access the referenced resources, in addition to the permission required to access the first resource. For example, to work with a web ACL that references an IP set, regex pattern set, or rule group, you need to have access to the IP set, regex pattern set, or rule group resource in addition to having access to the web ACL resource.

AWS WAF standard API permissions

The basic CRUD and list operations on AWS resources follow a standard pattern for permissions granting. The pattern applies to web ACLs, rule groups, IP sets, and regex pattern sets.

To grant permissions for Web ACLs

Apply the permissions for the web ACL and for any resource the web ACL references:

- Use the CRUD and list operations permissions guidance in this section for `WebACL` and `webacl`.
- For any rule groups that the web ACL references, use the guidance in this section with `RuleGroup` and `rulegroup`.
- For any managed rule groups that the web ACL references, provide the permissions for `DescribeManagedRuleGroup`, listed under AWS WAF non-standard API and required permissions for actions (p. 129).
- For any IP sets that the web ACL references, use the guidance in this section with `IPSet` and `ipset`.
- For any regex pattern sets that the web ACL references, use the guidance in this section with `RegexPatternSet` and `regexpatternset`.

To grant permissions for rule groups

Apply the permissions for the rule group and for any resource the rule group references:

- Use the CRUD and list operations permissions guidance in this section with `RuleGroup` and `rulegroup`.
- For any IP sets that the rule group references, use the guidance in this section with `IPSet` and `ipset`.
- For any regex pattern sets that the rule group references, use the guidance in this section with `RegexPatternSet` and `regexpatternset`.

To grant permissions for IP sets

For IP sets, use the CRUD and list operations permissions guidance in this section with `IPSet` and `ipset`.

To grant permissions for regex pattern sets

For regex pattern sets, use the CRUD and list operations permissions guidance in this section with `RegexPatternSet` and `regexpatternset`.

AWS WAF CRUD and List permissions

The patterns for CRUD and list apply to web ACLs, rule groups, IP sets, and regex pattern sets. This section shows the pattern for web ACL operations. For other resource types, substitute in the strings for those, according to the guidance preceding this section.

CRUD operations for web ACL

- **AWS WAF API Operations** – CreateWebACL, GetWebACL, UpdateWebACL, and DeleteWebACL
- **API Actions** – `wafv2:CreateWebACL`, `wafv2:GetWebACL`, `wafv2:UpdateWebACL`, `wafv2:DeleteWebACL`
- **Resources** – `arn:aws:wafv2:`*`region`*`:`*`account-id`*`:`*`scope`*`/webacl/`*`entity-name`*`/`*`entity-ID`*

List operations for web ACL

- **AWS WAF API Operation** – ListWebACLs
- **API Actions** – `wafv2:ListWebACLs`
- **Resources** – `arn:aws:wafv2:`*`region`*`:`*`account-id`*`:`*`scope`*`/`*`webacl`*`/*`

  If you want to list all resources in your account, call the list operation once for global, and once for each regional application region.

AWS WAF non-standard API permissions

The following operations don't follow the standard CRUD and list pattern and require specific resource permissions settings.

For each operation, we list the required policy actions and their associated policy resources.

AssociateWebACL

API Actions – `wafv2:AssociateWebACL`, `elasticloadbalancing:SetWebACL`, `apigateway:SetWebACL`, `appsync:SetWebACL`

Resources –

`arn:aws:wafv2:`*`region`*`:`*`account-id`*`:`*`scope`*`/webacl/`*`entity-name`*`/`*`entity-ID`*

`arn:aws:elasticloadbalancing:`*`region`*`:`*`account-id`*`:loadbalancer/app/`*`ApplicationLoadBalancerName`*`/`*`ApplicationLoadBalancerID`*

`arn:aws:apigateway:region::/restapis/api-ID/stages/stage-name`

`arn:aws:appsync:region:account-id:apis/GraphQLApiId`

CheckCapacity

API Action – `wafv2:CheckCapacity`

Resource – This requires permissions on all ARNs that are referenced in the contained rules. It doesn't require any other permissions.

DescribeManagedRuleGroup

API Action – `wafv2:DescribeManagedRuleGroup`

Resource – `arn:aws:wafv2:region:account-id:scope/managedruleset/*`

DisassociateWebACL

API Actions – `wafv2:DisassociateWebACL`, `elasticloadbalancing:SetWebACL`, `apigateway:SetWebACL`, `appsync:SetWebACL`

Resources –

`arn:aws:wafv2:region:account-id:scope/webacl/entity-name/entity-ID`

`arn:aws:elasticloadbalancing:region:account-id:loadbalancer/app/ApplicationLoadBalancerName/ApplicationLoadBalancerID`

`arn:aws:apigateway:region::/restapis/api-ID/stages/stage-name`

`arn:aws:appsync:region:account-id:apis/GraphQLApiId`

GetRateBasedStatementManagedKeys

API Action – `wafv2:GetRateBasedStatementManagedKeys`

Resource – `arn:aws:wafv2:region:account-id:scope/webacl/entity-name/entity-ID`

GetSampledRequests

API Action – `wafv2:GetSampledRequests`

Resource – The resource permissions depend on the parameters that you specify in the API call. You must have access to the web ACL that corresponds to the request for samples. For example: `arn:aws:wafv2:region:account-id:scope/webacl/entity-name/entity-ID`

ListAvailableManagedRuleGroups

API Action – `wafv2:ListAvailableManagedRuleGroups`

Resource – `arn:aws:wafv2:region:account-id:scope/managedruleset/*`

Using service-linked roles for AWS WAF

AWS WAF uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to AWS WAF. Service-linked roles are predefined by AWS WAF and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up AWS WAF easier because you don’t have to manually add the necessary permissions. AWS WAF defines the permissions of its service-linked roles, and unless defined otherwise, only AWS WAF can assume its roles. The defined permissions include the trust policy and the permissions policy. That permissions policy can't be attached to any other IAM entity.

You can delete a service-linked role only after first deleting the role's related resources. This protects your AWS WAF resources because you can't inadvertently remove permission to access the resources.

For information about other services that support service-linked roles, see AWS Services That Work with IAM and look for the services that have **Yes** in the **Service-Linked Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-linked role permissions for AWS WAF

AWS WAF uses the service-linked role `AWSServiceRoleForWAFV2Logging`.

AWS WAF uses this service-linked role to write logs to Amazon Kinesis Data Firehose. This role is used only if you enable logging in AWS WAF. For more information, see Logging web ACL traffic information (p. 107).

The `AWSServiceRoleForWAFV2Logging` service-linked role trusts the service to assume the role `wafv2.amazonaws.com`.

The permissions policies of the role allows AWS WAF to complete the following actions on the specified resources:

- Action: `firehose:PutRecord` and `firehose:PutRecordBatch` on Amazon Kinesis Data Firehose data stream resources with a name that starts with "aws-waf-logs-." For example, `aws-waf-logs-us-east-2-analytics`.

You must configure permissions to allow an IAM entity (such as a user, group, or role) to create, edit, or delete a service-linked role. For more information, see Service-Linked Role Permissions in the *IAM User Guide*.

Creating a service-linked role for AWS WAF

You don't need to manually create a service-linked role. When you enable AWS WAF logging on the AWS Management Console, or you make a `PutLoggingConfiguration` request in the AWS WAF CLI or the AWS WAF API, AWS WAF creates the service-linked role for you.

You must have the `iam:CreateServiceLinkedRole` permission to enable logging.

If you delete this service-linked role, and then need to create it again, you can use the same process to recreate the role in your account. When you enable AWS WAF logging, AWS WAF creates the service-linked role for you again.

Editing a service-linked role for AWS WAF

AWS WAF doesn't allow you to edit the `AWSServiceRoleForWAFV2Logging` service-linked role. After you create a service-linked role, you can't change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a Service-Linked Role in the *IAM User Guide*.

Deleting a service-linked role for AWS WAF

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don't have an unused entity that is not actively monitored or maintained. However, you must clean up the resources for your service-linked role before you can manually delete it.

> **Note**
> If the AWS WAF service is using the role when you try to delete the resources, then the deletion might fail. If that happens, wait for a few minutes and try the operation again.

To delete AWS WAF resources used by the `AWSServiceRoleForWAFV2Logging`

1. On the AWS WAF console, remove logging from every web ACL. For more information, see Logging web ACL traffic information (p. 107).
2. Using the API or CLI, submit a `DeleteLoggingConfiguration` request for each web ACL that has logging enabled. For more information, see AWS WAF API Reference.

To manually delete the service-linked role using IAM

Use the IAM console, the IAM CLI, or the IAM API to delete the `AWSServiceRoleForWAFV2Logging` service-linked role. For more information, see Deleting a Service-Linked Role in the *IAM User Guide*.

Supported Regions for AWS WAF service-linked roles

AWS WAF supports using service-linked roles in the following AWS Regions.

Region Name	Region Identity	Support in AWS WAF
US East (N. Virginia)	`us-east-1`	Yes
US East (Ohio)	`us-east-2`	Yes
US West (N. California)	`us-west-1`	Yes
US West (Oregon)	`us-west-2`	Yes
Asia Pacific (Mumbai)	`ap-south-1`	Yes
Asia Pacific (Osaka)	`ap-northeast-3`	Yes
Asia Pacific (Seoul)	`ap-northeast-2`	Yes
Asia Pacific (Singapore)	`ap-southeast-1`	Yes
Asia Pacific (Sydney)	`ap-southeast-2`	Yes
Asia Pacific (Tokyo)	`ap-northeast-1`	Yes
Canada (Central)	`ca-central-1`	Yes
Europe (Frankfurt)	`eu-central-1`	Yes
Europe (Ireland)	`eu-west-1`	Yes
Europe (London)	`eu-west-2`	Yes
Europe (Paris)	`eu-west-3`	Yes
South America (São Paulo)	`sa-east-1`	Yes

Logging and monitoring in AWS WAF

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS WAF and your AWS solutions. You should collect monitoring data from all parts of your AWS solution so that you can more easily debug a multi-point failure if one occurs. AWS provides several tools for monitoring your AWS WAF resources and responding to potential events:

Amazon CloudWatch alarms

Using CloudWatch alarms, you watch a single metric over a time period that you specify. If the metric exceeds a given threshold, CloudWatch sends a notification to an Amazon SNS topic or AWS Auto Scaling policy. For more information, see Monitoring with Amazon CloudWatch (p. 370).

AWS CloudTrail logs

CloudTrail provides a record of actions taken by a user, role, or an AWS service in AWS WAF. Using the information collected by CloudTrail, you can determine the request that was made to AWS WAF, the IP address from which the request was made, who made the request, when it was made, and additional details. For more information, see Logging API calls with AWS CloudTrail (p. 376).

AWS WAF web ACL traffic logging

AWS WAF offers logging for the traffic that your web ACLs analyze. The logs include information such as the time that AWS WAF received the request from your protected AWS resource, detailed information about the request, and the action setting for the rule that the request matched. For more information, see Logging web ACL traffic information (p. 107).

Compliance validation for AWS WAF

Third-party auditors assess the security and compliance of AWS WAF as part of multiple AWS compliance programs, such as SOC, PCI, FedRAMP, HIPAA, and others.

For a list of AWS services in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS WAF is determined by the sensitivity of your data, your organization's compliance objectives, and applicable laws and regulations. If your use of AWS WAF is subject to compliance with standards like HIPAA, PCI, or FedRAMP, AWS provides resources to help:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- Architecting for HIPAA Security and Compliance Whitepaper – This technical paper describes how companies can use AWS to create HIPAA-compliant applications.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- AWS Config – This AWS service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.
- AWS Well-Architected Framework – The AWS Well-Architected Framework helps you build secure cloud applications.

Resilience in AWS WAF

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

Infrastructure security in AWS WAF

As a managed service, AWS WAF is protected by the AWS global network security procedures that are described in Amazon Web Services: Overview of Security Processes.

You use AWS published API calls to access AWS WAF through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

AWS WAF quotas

Note
This is the latest version of AWS WAF. For AWS WAF Classic, see AWS WAF Classic (p. 138).

AWS WAF is subject to the following quotas (formerly referred to as limits). These quotas are the same for all Regions in which AWS WAF is available. Each Region is subject to these quotas individually. The quotas are not cumulative across Regions.

AWS WAF has default quotas on the maximum number of entities you can have per account. You can request an increase in these quotas.

Resource	Default quota per account per Region
Maximum number of web ACLs	100
Maximum number of rule groups	100
Maximum web ACL capacity units (WCUs) per web ACL	1,500
Maximum WCUs per rule group	1,500
Maximum number of IP sets	100
Maximum number of requests per second per web ACL (applies only to Application Load Balancers)	25,000
Maximum number of custom request headers per web ACL or rule group	100

Resource	Default quota per account per Region
Maximum number of custom response headers per web ACL or rule group	100
Maximum number of custom response bodies per web ACL or rule group	50

The maximum requests per second (RPS) allowed for AWS WAF on CloudFront is set by CloudFront and described in the CloudFront Developer Guide.

AWS WAF has fixed quotas on the following entity settings per account per Region. These quotas can't be changed.

Resource	Quota per account per Region
Maximum number of references per rule group to IP sets and regex pattern sets	50
Maximum number of references per web ACL to IP sets, regex pattern sets, and rule groups	50
Maximum number of IP addresses in CIDR notation per IP set	10,000
Maximum number of rate-based rules per web ACL	10
Maximum number of rate-based rules per rule group	4
Maximum number of unique IP addresses that can be blocked per rate-based rule	10,000
Maximum number of characters in a string match statement	200
Maximum number of characters in each regex pattern	200
Maximum number of unique regex patterns per regex set	10
Maximum number of regex sets	10
Maximum size of a web request body that can be inspected	8 KB
Minimum request rate that can be defined for a rate-based rule	100
Maximum number of text transformations per rule statement	3
Maximum size of the custom response body content for a single custom response definition	4 KB
Maximum number of custom headers for a single custom response definition	10
Maximum number of custom headers for a single custom request definition	10
Maximum combined size of all response body content for a single rule group or a single web ACL	50 KB

AWS WAF has the following fixed quotas on calls per account per Region. These quotas apply to the total calls to the service through any available means, including the console, CLI, AWS CloudFormation, the REST API, and the SDKs. These quotas can't be changed.

Call type	Quota per account per Region
Maximum number of calls to `AssociateWebACL`	One request every 2 seconds
Maximum number of calls to `DisassociateWebACL`	One request every 2 seconds
Maximum number of calls to `GetWebACLForResource`	One request per second
Maximum number of calls to `ListResourcesForWebACL`	One request per second
Maximum number of calls to any individual `Get` or `List` action, if no other quota is defined for it	Five requests per second
Maximum number of calls to any individual `Create`, `Put`, or `Update` action, if no other quota is defined for it	One request per second

AWS WAF Classic

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

AWS WAF Classic is a web application firewall that lets you monitor the HTTP and HTTPS requests that are forwarded to an Amazon API Gateway API, Amazon CloudFront or an Application Load Balancer. AWS WAF Classic 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, API Gateway, 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

Setting up AWS WAF Classic

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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

Note
If you are a new user, don't follow these setup steps for AWS WAF Classic. Instead, follow the steps for the latest version of AWS WAF, at Setting up (p. 3).

After you complete these steps, see Getting started with AWS WAF Classic (p. 144) to continue getting started with AWS WAF Classic.

> **Note**
> AWS Shield Standard is included with AWS WAF Classic and does not require additional setup. For more information, see How AWS Shield works (p. 325).

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

- Step 1: Sign up for an AWS account (p. 139)
- Step 2: Create an IAM user (p. 139)
- Step 3: Download tools (p. 141)

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 Classic. 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://portal.aws.amazon.com/billing/signup.
2. Follow the online instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a verification code on 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 Classic console, you must sign in to confirm that you have permission to perform AWS WAF Classic 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 administrative permissions directly to the IAM user account.
- Verify that the account has full access to AWS WAF Classic and related services, for general use and for console access. For information, see AWS managed (predefined) policies for AWS WAF Classic (p. 229).

You then can sign in to the AWS WAF Classic 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 Classic by using the AWS Command Line Interface (AWS CLI), Tools for Windows PowerShell, the AWS SDKs, or the AWS WAF Classic 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 administrator user for yourself and add the user to an administrators group (console)

1. Sign in to the IAM console as the account owner by choosing **Root user** and entering your AWS account email address. On the next page, enter your password.

   Note
   We strongly recommend that you adhere to the best practice of using the **`Administrator`** IAM user that follows and securely lock away the root user credentials. Sign in as the root user only to perform a few account and service management tasks.
2. In the navigation pane, choose **Users** and then choose **Add user**.
3. For **User name**, enter **`Administrator`**.
4. Select the check box next to **AWS Management Console access**. Then select **Custom password**, and then enter your new password in the text box.
5. (Optional) By default, AWS requires the new user to create a new password when first signing in. You can clear the check box next to **User must create a new password at next sign-in** to allow the new user to reset their password after they sign in.
6. Choose **Next: Permissions**.
7. Under **Set permissions**, choose **Add user to group**.
8. Choose **Create group**.
9. In the **Create group** dialog box, for **Group name** enter **`Administrators`**.
10. Choose **Filter policies**, and then select **AWS managed - job function** to filter the table contents.
11. In the policy list, select the check box for **AdministratorAccess**. Then choose **Create group**.

    Note
    You must activate IAM user and role access to Billing before you can use the `AdministratorAccess` permissions to access the AWS Billing and Cost Management console. To do this, follow the instructions in step 1 of the tutorial about delegating access to the billing console.
12. 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.
13. Choose **Next: Tags**.
14. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information about using tags in IAM, see Tagging IAM entities in the *IAM User Guide*.
15. 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 that restrict user permissions to specific AWS resources, see Access management and Example policies.

To sign in as this new IAM user, first sign out of the AWS Management 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.

After you complete these steps, you can stop here and go to Getting started with AWS WAF Classic (p. 144) to continue getting started with AWS WAF Classic using the console. If you want to access AWS WAF Classic programmatically using the AWS WAF Classic API, continue on to the next step, Step 3: Download tools (p. 141).

Step 3: Download tools

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

- If you want to call the AWS WAF Classic 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 Classic 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 Classic supports.
- The AWS Command Line Interface (AWS CLI) supports AWS WAF Classic. 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 Classic. For more information, see AWS Tools for PowerShell Cmdlet Reference.

How AWS WAF Classic works

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

You use AWS WAF Classic to control how API Gateway, 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.

Note
You can also use AWS WAF Classic to protect your applications that are hosted in Amazon Elastic Container Service (Amazon ECS) containers. Amazon ECS is a highly scalable, fast container management service that makes it easy to run, stop, and manage Docker containers on a cluster. To use this option, you configure Amazon ECS to use an AWS WAF Classic enabled Application Load Balancer to route and protect HTTP/HTTPS (layer 7) traffic across the tasks in your service. For more information, see the topic Service Load Balancing in the *Amazon Elastic Container Service Developer Guide*.

Conditions

Conditions define the basic characteristics that you want AWS WAF Classic 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.
- Country or geographical location 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. You can also use regular expressions (regex) to specify these strings.

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 Classic 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 Classic looks for requests that match all conditions—that is, it `AND`s the conditions together.

Add at least one condition to a regular rule. A regular rule without conditions can't match any requests, so the rule's action (allow, count, or block) is never triggered.

Rate-based rule

Rate-based rules are like regular rules with an added rate limit. A rate-based rule counts the requests that arrive from IP addresses that satisfy the rule's conditions. If the requests from an IP address exceed the rate limit in a five-minute period, the rule can trigger an action. It can take a minute or two for the action to trigger.

Conditions are optional for rate-based rules. If you don't add any conditions in a rate-based rule, the rate limit applies to all IP addresses. If you combine conditions with the rate limit, the rate limit applies to IP addresses that match the conditions.

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 1,000. Requests that meet both of the preceding conditions and exceed 1,000 requests per five minutes trigger the rule's action (block or count), which is defined in the web ACL.

Requests that don't meet both conditions aren't counted towards the rate limit and aren't affected 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 1,000.

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.

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 Classic can either block the request or allow the request to be forwarded to the API Gateway API, CloudFront distribution or an Application Load Balancer. You specify the action that you want AWS WAF Classic to perform for each rule.

AWS WAF Classic compares a request with the rules in a web ACL in the order in which you listed the rules. AWS WAF Classic 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 Classic 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 Classic 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 Classic 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 Classic forwards the request to API Gateway, 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 Classic 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 Classic 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 some occasions, AWS WAF might encounter an internal error that delays the response to Amazon API Gateway, Amazon CloudFront or an Application Load Balancer about whether to allow or block a request. On those occasions CloudFront will typically allow the request or serve the content. API Gateway and an Application Load Balancer typically will deny the request and not serve the content.

AWS WAF Classic pricing

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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

Getting started with AWS WAF Classic

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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

- Set up AWS WAF Classic.
- Create a web access control list (web ACL) using the AWS WAF Classic 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 Classic 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 Classic takes when a web request doesn't match any of your rules.
- Choose the Amazon CloudFront distribution that you want AWS WAF Classic to inspect web requests for. This tutorial covers the steps only for CloudFront, but the process for an Application Load Balancer and Amazon API Gateway APIs essentially is the same. AWS WAF Classic for CloudFront is available for all AWS Regions. AWS WAF Classic for use with API Gateway or an Application Load Balancer is available in the Regions listed at AWS service endpoints.

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 Classic

If you already signed up for an AWS account and created an IAM user as described in Setting up AWS WAF Classic (p. 138), go to Step 2: Create a Web ACL (p. 145).

If not, go to Setting up AWS WAF Classic (p. 138) 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 Classic console.)

Step 2: Create a Web ACL

The AWS WAF Classic console guides you through the process of configuring AWS WAF Classic 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/wafv2/.
2. If this is your first time using AWS WAF Classic, choose **Go to AWS WAF Classic**, and then choose **Configure web ACL**.

   If you've used AWS WAF Classic 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**, enter a name.

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

   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. 167).

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**, enter a name. The name can contain only alphanumeric characters (A-Z, a-z, 0-9) or the following special characters: _-!"# ` +*},./ .
3. For **Address**, enter **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.)

   AWS WAF Classic supports IPv4 address ranges: /8 and any range between /16 through /32. AWS WAF Classic supports IPv6 address ranges: /24, /32, /48, /56, /64, and /128. (To specify a single IP address, such as 192.0.2.44, enter **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 geo match condition

A geo match condition specifies the country or countries that requests originate from. In this step, you create a geo match condition. In a later step, you specify whether you want to allow requests or block requests that originate from the specified countries.

> **Note**
> For more information about geo match conditions, see Working with geographic match conditions (p. 169).

To create a geo match condition

1. On the **Create conditions** page, for **Geo match conditions**, choose **Create condition**.
2. In the **Create geo match condition** dialog box, for **Name**, enter a name. The name can contain only alphanumeric characters (A-Z, a-z, 0-9) or the following special characters: _-!"# ` +*},./ .
3. Choose a **Location type** and a country. Currently, **Location type** can only be **Country**.
4. Choose **Add location**.
5. Choose **Create**.

Step 5: Create a string match condition

A string match condition identifies the strings that you want AWS WAF Classic 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. 180).

To create a string match condition

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

Name

Enter a name. The name can contain only alphanumeric characters (A-Z, a-z, 0-9) or the following special characters: _-!"#`+*},./ .

Type

Choose **String match**.

Part of the request to filter on

Choose the part of the web request that you want AWS WAF Classic 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 Classic 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 Classic gets the length of the body from the request headers.) For more information, see Working with size constraint conditions (p. 171).

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 Classic to inspect. Enter **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 Classic inspects web requests for a header value that is identical to the value that you specify.

Transformation

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

You can only specify a single type of text transformation.

For this example, choose **None**.

Value is base64 encoded

When the value that you enter 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 Classic to search for in the part of web requests that you indicated in **Part of the request to filter on**.

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

The maximum length of **Value to match** is 50 characters. If you want to specify a base64-encoded value, you can provide up to 50 characters before encoding.

3. If you want AWS WAF Classic 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 5A: Create a regex condition (optional)

A regular expression condition is a type of string match condition and similar in that it identifies the strings that you want AWS WAF Classic to search for in a request, such as a specified value in a header or in a query string. The primary difference is that you use a regular expression (regex) to specify the string pattern that you want AWS WAF Classic to search for. In this step, you create a regex 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 regex match conditions, see Working with regex match conditions (p. 185).

To create a regex match condition

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

   Name

   Enter a name. The name can contain only alphanumeric characters (A-Z, a-z, 0-9) or the following special characters: _-!"#`+*},./ .

   Type

   Choose **Regex match**.

   Part of the request to filter on

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

   For this example, choose **Body**.

   Note
   If you choose **Body** for the value of **Part of the request to filter on**, AWS WAF Classic 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 Classic gets the length of the body from the request headers.) For more information, see Working with size constraint conditions (p. 171).

Transformation

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

You can only specify a single type of text transformation.

For this example, choose **None**.

Regex patterns to match to request

Choose **Create regex pattern set**.

New pattern set name

Enter a name and then specify the regex pattern that you want AWS WAF Classic to search for.

Next, enter the regular expression **I[a@]mAB[a@]dRequest**. AWS WAF Classic will inspect the `User-Agent` header in web requests for the values:

- **IamABadRequest**
- **IamAB@dRequest**
- **I@mABadRequest**
- **I@mAB@dRequest**

3. Choose **Create pattern set and add filter**.
4. Choose **Create**.

Step 6: 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 Classic 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. 175).

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, enter the following values:

Name

Enter a name.

Part of the request to filter on

Choose the part of web requests that you want AWS WAF Classic 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 Classic 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 Classic gets the length of the body from the request headers.) For more information, see Working with size constraint conditions (p. 171).

Transformation

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

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

You can only specify a single type of text transformation.

3. Choose **Create**.
4. Choose **Next**.

Step 7: (Optional) create additional conditions

AWS WAF Classic 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 Classic to check for length. For more information, see Working with size constraint conditions (p. 171).
- **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. 162).

You can optionally create these conditions now, or you can skip to Step 8: Create a rule and add conditions (p. 150).

Step 8: Create a rule and add conditions

You create a rule to specify the conditions that you want AWS WAF Classic 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 Classic to allow or block requests based on that rule.

Note
For more information about rules, see Working with rules (p. 190).

To create a rule and add conditions

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

Name

Enter a name.

CloudWatch metric name

Enter a name for the CloudWatch metric that AWS WAF Classic 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 white space.

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 in any five-minute period. For more information about the rule types, see How AWS WAF Classic works (p. 141). For this example, choose `Regular rule`.

Rate limit

For a rate-based rule, enter the maximum number of requests to allow in any five-minute period from an IP address that matches the rule's conditions.

3. For the first condition that you want to add to the rule, specify the following settings:
   - Choose whether you want AWS WAF Classic 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 condition**.
5. Add the geo match condition that you created earlier. Specify the following values:
   - **When a request does**
   - **originate from a geographic location in**
   - Choose your geo match condition.
6. Choose **Add another condition**.
7. 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.
8. Choose **Add condition**.
9. 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.
10. Choose **Add condition**.
11. 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.
12. If you created any other conditions, such as a regex condition, add those in a similar manner.
13. Choose **Create**.
14. For the **Default action**, choose **Allow all requests that don't match any rules**.
15. Choose **Review and create**.

Step 9: 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 Classic 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 Classic to take on web requests that *do not* match all the conditions in the rule: allow or block the requests.

AWS WAF Classic starts blocking CloudFront web requests that match all the following conditions (and any others you might have added):

- The value of the `User-Agent` header is `BadBot`
- (If you created and added the regex condition) The value of the `Body` is any of the four strings that matches the pattern `I[a@]mAB[a@]dRequest`
- The requests originate from IP addresses in the range 192.0.2.0-192.0.2.255
- The requests originate from the country that you selected in your geo match condition
- The requests appear to include malicious SQL code in the query string

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

Step 10: Clean up your resources

You've now successfully completed the tutorial. To prevent your account from accruing additional AWS WAF Classic charges, you should clean up the AWS WAF Classic 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 Classic 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/wafv2/.
   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 Classic 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 Classic 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 and regex 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. If you created one, delete the filter in your regex match condition, and delete the regex match condition:

a. In the navigation pane, choose **String and regex matching**.
b. Choose the regex 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 **Regex match conditions** pane, choose **Delete**.

f. In the **Delete** dialog box, choose **Delete** again to confirm.

5. 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 for AWS WAF Classic

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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 Classic. The tutorials show how to combine several AWS services to automatically configure AWS WAF Classic 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 Classic Preconfigured Protections

You can use our preconfigured template to get started quickly with AWS WAF Classic. The template includes a set of AWS WAF Classic 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 Classic begins to block the web requests to your CloudFront distribution or to an Application Load Balancer that matches 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 Classic Security Automations.

Tutorial: Quickly setting up AWS WAF Classic protection against common attacks

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

This tutorial shows you how to use AWS CloudFormation to quickly configure AWS WAF Classic 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 Classic 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 Classic 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 Classic 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. 158) 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 Classic 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 Classic 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. 159).

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. 158) later in this tutorial.

***prefix*SizeMatchRule**

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

***prefix*SqliRule**

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

***prefix*XssRule**

AWS WAF Classic 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 Classic Pricing and Amazon CloudFront Pricing.

Step 1: Create an AWS CloudFormation stack that sets up AWS WAF Classic protection against common attacks

In the following procedure, you use an AWS CloudFormation template to create a stack that sets up AWS WAF Classic 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. 159).

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 Europe (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 **Specify template** page, choose **Amazon S3 URL**. For the template URL, type **`https://s3.amazonaws.com/cloudformation-examples/community/common-attacks.json`**.
4. Choose **Next**.
5. On the **Specify stack 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 **Configure stack 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 stack**.

   After you choose **Create stack**, AWS CloudFormation creates the AWS WAF Classic resources that are identified in Solution overview (p. 155).

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 Classic.

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/wafv2/.
2. Choose **Go to AWS WAF Classic**.
3. In the navigation pane, choose **Web ACLs**.
4. Choose the web ACL that you want to associate with a CloudFront distribution.
5. On the **Rules** tab, under **AWS resources using this web ACL**, choose **Add association**.
6. When prompted, use the **Resource** list to choose the distribution that you want to associate this web ACL with.
7. Choose **Add**.
8. To associate this web ACL with additional CloudFront distributions, repeat steps 4 through 6.

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/wafv2/.
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 Classic supports IPv4 address ranges: /8 and any range between /16 through /32. AWS WAF Classic supports IPv6 address ranges: /24, /32, /48, /56, /64, and /128. For more information about CIDR notation, see the Wikipedia entry Classless Inter-Domain Routing.

      Note
      AWS WAF Classic 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/wafv2/.
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. 155), delete the AWS CloudFormation stack that you created in Step 1: Create an AWS CloudFormation stack that sets up AWS WAF Classic protection against common attacks (p. 157). This deletes the AWS WAF Classic 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 Classic samples, including Lambda functions, AWS CloudFormation templates, and SDK usage examples, go to GitHub at https://github.com/awslabs/aws-waf-sample.

Blog tutorials

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

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)

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

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

- Originate from an IP address or a range of IP addresses
- Originate from a specific country or countries
- Contain a specified string or match a regular expression (regex) pattern 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. 197).
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. 162).
   - 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. 167).
   - To allow or block requests based on the country that they originate from, create geo match conditions. For more information, see Working with geographic match conditions (p. 169).
   - 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. 171).
   - 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. 175).
   - 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. 180).
   - To allow or block requests based on a regex pattern that appear in the requests, create regex match conditions. For more information, see Working with regex match conditions (p. 185).
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 Classic to allow or block requests based on the rule. For more information, see Working with rules (p. 190). Optionally, you can use a rate-based rule instead of a regular rule to limit the number of requests from any IP address that meets the conditions.
4. Add the rules to a web ACL. For each rule, specify whether you want AWS WAF Classic 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 Classic evaluates the rules in the order that they're listed in the web ACL. For more information, see Working with web ACLs (p. 197).

   When you add a new rule or update existing rules, it can take up to one minute for those changes to appear and be active across your web ACLs and resources.

Topics

Working with conditions

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not

migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

Conditions specify when 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. 162).
- 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. 167).
- To allow or block requests based on the country that they originate from, create geo match conditions. For more information, see Working with geographic match conditions (p. 169).
- 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. 171).
- 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. 175).
- 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. 180).
- To allow or block requests based on a regex pattern that appear in the requests, create regex match conditions. For more information, see Working with regex match conditions (p. 185).

Topics

Working with cross-site scripting match conditions

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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 Classic 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. The filters indicate the part of web requests that you want AWS WAF Classic 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 Classic 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 Classic 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 Classic to inspect the URI for malicious scripts, and the other instructs AWS WAF Classic to inspect the query string. AWS WAF Classic 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 Classic 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 Classic 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 Classic 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/wafv2/.
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. 163).
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, 0-9, and the special characters: _-!"#`+*},./ . 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 Classic 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.

> **Note**
> For cross-site scripting match conditions, we recommend that you choose **All query parameters (values only)** instead of **Query string** for **Part of the request to filter on**.

URI

The URI path of the request, which identifies the resource, for example, `/images/daily-ad.jpg`. This doesn't include the query string or fragment components of the URI. For information, see Uniform Resource Identifier (URI): Generic Syntax.

Unless a **Transformation** is specified, a URI is not normalized and is inspected just as AWS receives it from the client as part of the request. A **Transformation** will reformat the URI as specified.

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 Classic 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 Classic gets the length of the body from the request headers.) For more information, see Working with size constraint conditions (p. 171).

Single query parameter (value only)

Any parameter that you have defined as part of the query string. For example, if the URL is "www.xyz.com?UserName=abc&SalesRegion=seattle" you can add a filter to either the *UserName* or *SalesRegion* parameter.

If you choose **Single query parameter (value only)**, you will also specify a **Query parameter name**. This is the parameter in the query string that you will inspect, such as *UserName* or *SalesRegion*. The maximum length for **Query parameter name** is 30 characters. **Query parameter name** is not case sensitive. For example, it you specify *UserName* as the **Query parameter name**, this will match all variations of *UserName*, such as *username* and *UsERName*.

All query parameters (values only)

Similar to **Single query parameter (value only)**, but rather than inspecting the values of a single parameter, AWS WAF Classic inspects all parameter values within the query string for possible malicious scripts. For example, if the URL is "www.xyz.com?UserName=abc&SalesRegion=seattle," and you choose **All query parameters (values only)**, AWS WAF Classic will trigger a match if either the value of *UserName* or *SalesRegion* contain possible malicious scripts.

Header

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

Transformation

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

You can only specify a single type of text transformation.

Transformations can perform the following operations:

None

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

HTML decode

AWS WAF Classic replaces HTML-encoded characters with unencoded characters:

- Replaces `&quot;` with `&`
- Replaces ` ` with a non-breaking space
- Replaces `<` with <
- Replaces `>` 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 white space

AWS WAF Classic 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/wafv2/.
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. 163).
   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/wafv2/.
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

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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 Classic 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/wafv2/.
2. In the navigation pane, choose **IP addresses**.
3. Choose **Create condition**.
4. Enter a name in the **Name** field.

   The name can contain only alphanumeric characters (A-Z, a-z, 0-9) or the following special characters: _-!"# `+*},./ . 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, enter **2620:0:2d0:200::/64**.

   AWS WAF Classic supports IPv4 address ranges: /8 and any range between /16 through /32. AWS WAF Classic supports IPv6 address ranges: /24, /32, /48, /56, /64, and /128. 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/wafv2/.
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 enter an IP address range by using CIDR notation. Here are some examples:

   - To specify the IPv4 address 192.0.2.44, enter **192.0.2.44/32**.
   - To specify the IPv6 address 0:0:0:0:0:ffff:c000:22c, enter **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, enter **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, enter **2620:0:2d0:200::/64**.

   AWS WAF Classic supports IPv4 address ranges: /8 and any range between /16 through /32. AWS WAF Classic supports IPv6 address ranges: /24, /32, /48, /56, /64, and /128. 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 enter 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/wafv2/.
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 geographic match conditions

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

If you want to allow or block web requests based on the country that the requests originate from, create one or more geo match conditions. A geo match condition lists countries 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 countries.

You can use geo match conditions with other AWS WAF Classic conditions or rules to build sophisticated filtering. For example, if you want to block certain countries, but still allow specific IP addresses from that country, you could create a rule containing a geo match condition and an IP match condition. Configure the rule to block requests that originate from that country and do not match the approved IP addresses. As another example, if you want to prioritize resources for users in a particular country, you could include a geo match condition in two different rate-based rules. Set a higher rate limit for users in the preferred country and set a lower rate limit for all other users.

Note
If you are using the CloudFront geo restriction feature to block a country from accessing your content, any request from that country is blocked and is not forwarded to AWS WAF Classic. So if you want to allow or block requests based on geography plus other AWS WAF Classic conditions, you should *not* use the CloudFront geo restriction feature. Instead, you should use an AWS WAF Classic geo match condition.

Topics

Creating a geo match condition

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

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

To create a geo match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **Geo match**.
3. Choose **Create condition**.
4. Enter a name in the **Name** field.

   The name can contain only alphanumeric characters (A-Z, a-z, 0-9) or the following special characters: _-!"# `+*},./ . You can't change the name of a condition after you create it.
5. Choose a **Region**.
6. Choose a **Location type** and a country. **Location type** can currently only be **Country**.
7. Choose **Add location**.
8. Choose **Create**.

Editing geo match conditions

You can add countries to or delete countries from your geo match condition.

To edit a geo match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **Geo match**.
3. In the **Geo match conditions** pane, choose the geo match condition that you want to edit.
4. To add a country:

   a. In the right pane, choose **Add filter**.
   b. Choose a **Location type** and a country. **Location type** can currently only be **Country**.
   c. Choose **Add**.
5. To delete a country:

   a. In the right pane, select the values that you want to delete.
   b. Choose **Delete filter**.

Deleting geo match conditions

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

To delete a geo match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. Remove the geo match condition from the rules that are using it:

   a. In the navigation pane, choose **Rules**.
   b. Choose the name of a rule that is using the geo match condition that you want to delete.
   c. In the right pane, choose **Edit rule**.
   d. Choose the **X** next to the condition you want to delete.
   e. Choose **Update**.

   f. Repeat for all the remaining rules that are using the geo match condition that you want to delete.
3. Remove the filters from the condition you want to delete:
   a. In the navigation pane, choose **Geo match**.
   b. Choose the name of the geo match condition that you want to delete.
   c. In the right pane, choose the check box next to **Filter** in order to select all of the filters.
   d. Choose the **Delete filter**.
4. In the navigation pane, choose **Geo match**.
5. In the **Geo match conditions** pane, choose the geo match condition that you want to delete.
6. Choose **Delete** to delete the selected condition.

Working with size constraint conditions

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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 Classic to look at, the number of bytes that you want AWS WAF Classic 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 Classic to inspect the request body, for example, by searching the body for a specified string, AWS WAF Classic 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 Classic 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 Classic 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 Classic 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 Classic 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 Classic 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 Classic 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 Classic 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/wafv2/.
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. 172).
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

Enter a name for the size constraint condition.

The name can contain only alphanumeric characters (A-Z, a-z, 0-9) or the following special characters: _-!"#`+*},./. 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 Classic 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 URI path of the request, which identifies the resource, for example, `/images/daily-ad.jpg`. This doesn't include the query string or fragment components of the URI. For information, see Uniform Resource Identifier (URI): Generic Syntax.

Unless a **Transformation** is specified, a URI is not normalized and is inspected just as AWS receives it from the client as part of the request. A **Transformation** will reformat the URI as specified.

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.

Single query parameter (value only)

Any parameter that you have defined as part of the query string. For example, if the URL is "www.xyz.com?UserName=abc&SalesRegion=seattle" you can add a filter to either the *UserName* or *SalesRegion* parameter.

If you choose **Single query parameter (value only)**, you will also specify a **Query parameter name**. This is the parameter in the query string that you will inspect, such as *UserName*. The maximum length for **Query parameter name** is 30 characters. **Query parameter name** is not case sensitive. For example, it you specify *UserName* as the **Query parameter name**, this will match all variations of *UserName*, such as *username* and *UsERName*.

All query parameters (values only)

Similar to **Single query parameter (value only)**, but rather than inspecting the value of a single parameter, AWS WAF Classic inspects the values of all parameters within the query string for the size constraint. For example, if the URL is "www.xyz.com?UserName=abc&SalesRegion=seattle," and you choose **All query parameters (values only)**, AWS WAF Classic will trigger a match the value of if either *UserName* or *SalesRegion* exceed the specified size.

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 Classic to evaluate the length.

Comparison operator

Choose how you want AWS WAF Classic 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 Classic evaluates web requests for a query string that is longer than 100 bytes.

Size

Enter the length, in bytes, that you want AWS WAF Classic 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 path `/logo.jpg` is nine characters long.

Transformation

A transformation reformats a web request before AWS WAF Classic 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 Classic.

> **Note**
> If you choose **Body** for **Part of the request to filter on**, you can't configure AWS WAF Classic 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 Classic gets the length of the body from the request headers.)

You can only specify a single type of text transformation.

Transformations can perform the following operations:

None

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

Convert to lowercase

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

HTML decode

AWS WAF Classic replaces HTML-encoded characters with unencoded characters:

- Replaces `&quot;` with `&`
- Replaces ` ` with a non-breaking space
- Replaces `<` with <
- Replaces `>` 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 white space

AWS WAF Classic 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/wafv2/.
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. 172).
   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/wafv2/.
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

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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 path or the query string, that you want AWS WAF Classic 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 Classic 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 Classic 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 Classic 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 Classic to inspect the URI for malicious SQL code, and the other instructs AWS WAF Classic to inspect the query string. AWS WAF Classic 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 Classic 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 Classic 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 Classic 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/wafv2/.
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. 176).
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 alphanumeric characters (A-Z, a-z, 0-9) or the following special characters: _-!"#`+*},./. 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 Classic 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.

> **Note**
> For SQL injection match conditions, we recommend that you choose **All query parameters (values only)** instead of **Query string** for **Part of the request to filter on**.

URI

The URI path of the request, which identifies the resource, for example, `/images/daily-ad.jpg`. This doesn't include the query string or fragment components of the URI. For information, see Uniform Resource Identifier (URI): Generic Syntax.

Unless a **Transformation** is specified, a URI is not normalized and is inspected just as AWS receives it from the client as part of the request. A **Transformation** will reformat the URI as specified.

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 Classic 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 Classic gets the length of the body from the request headers.) For more information, see Working with size constraint conditions (p. 171).

Single query parameter (value only)

Any parameter that you have defined as part of the query string. For example, if the URL is "www.xyz.com?UserName=abc&SalesRegion=seattle" you can add a filter to either the *UserName* or *SalesRegion* parameter.

If you choose **Single query parameter (value only)** you will also specify a **Query parameter name**. This is the parameter in the query string that you will inspect, such as *UserName* or *SalesRegion*. The maximum length for **Query parameter name** is 30 characters. **Query parameter name** is not case sensitive. For example, it you specify *UserName* as the **Query parameter name**, this will match all variations of *UserName*, such as *username* and *UsERName*.

All query parameters (values only)

Similar to **Single query parameter (value only)**, but rather than inspecting the value of a single parameter, AWS WAF Classic inspects the value of all parameters within the query string for possible malicious SQL code. For example, if the URL is "www.xyz.com?UserName=abc&SalesRegion=seattle," and you choose **All query parameters (values only)**, AWS WAF Classic will trigger a match if the value of either *UserName* or *SalesRegion* contain possible malicious SQL code.

Header

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

Transformation

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

You can only specify a single type of text transformation.

Transformations can perform the following operations:

None

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

HTML decode

AWS WAF Classic replaces HTML-encoded characters with unencoded characters:

- Replaces `&quot;` with `&`
- Replaces ` ` with a non-breaking space
- Replaces `<` with <
- Replaces `>` 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 white space

AWS WAF Classic 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/wafv2/.
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. 176).
   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/wafv2/.
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

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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 Classic 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 Classic 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 Classic 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 Classic 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 Classic 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 Classic 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 Classic 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 Classic 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/wafv2/.
2. In the navigation pane, choose **String and regex 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. 181).
5. Choose **Add 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 string match conditions

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

Name

Enter a name for the string match condition. The name can contain only alphanumeric characters (A-Z, a-z, 0-9) or the following special characters: _-!"#`+*},./. You can't change the name of a condition after you create it.

Type

Choose **String match**.

Part of the request to filter on

Choose the part of each web request that you want AWS WAF Classic 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 URI path of the request, which identifies the resource, for example, `/images/daily-ad.jpg`. This doesn't include the query string or fragment components of the URI. For information, see Uniform Resource Identifier (URI): Generic Syntax.

Unless a **Transformation** is specified, a URI is not normalized and is inspected just as AWS receives it from the client as part of the request. A **Transformation** will reformat the URI as specified.

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 Classic 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 Classic gets the length of the body from the request headers.) For more information, see Working with size constraint conditions (p. 171).

Single query parameter (value only)

Any parameter that you have defined as part of the query string. For example, if the URL is "www.xyz.com?UserName=abc&SalesRegion=seattle" you can add a filter to either the *UserName* or *SalesRegion* parameter.

If duplicate parameters appear in the query string, the values are evaluated as an "OR." That is, either value will trigger a match. For example, in the URL "www.xyz.com?SalesRegion=boston&SalesRegion=seattle", either "boston" or "seattle" in **Value to match** will trigger a match.

If you choose **Single query parameter (value only)** you will also specify a **Query parameter name**. This is the parameter in the query string that you will inspect, such as *UserName* or *SalesRegion*. The maximum length for **Query parameter name** is 30 characters. **Query parameter name** is not case sensitive. For example, it you specify *UserName* as the **Query parameter name**, this will match all variations of *UserName*, such as *username* and *UsERName*.

All query parameters (values only)

Similar to **Single query parameter (value only)**, but rather than inspecting the value of a single parameter, AWS WAF Classic inspects the value of all parameters within the query string for the **Value to match**. For example, if the URL is "www.xyz.com?UserName=abc&SalesRegion=seattle," and you choose **All query parameters (values only)**, AWS WAF Classic will trigger a match if the value of either *UserName* or *SalesRegion* is specified as the **Value to match**.

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 enter the name of a header that you want AWS WAF Classic to inspect.

Match type

Within the part of the request that you want AWS WAF Classic 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 Classic inspects the request. This eliminates some of the unusual formatting that attackers use in web requests in an effort to bypass AWS WAF Classic.

You can only specify a single type of text transformation.

Transformations can perform the following operations:

None

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

HTML decode

AWS WAF Classic replaces HTML-encoded characters with unencoded characters:

- Replaces `&quot;` with `&`
- Replaces ` ` with a non-breaking space
- Replaces `<` with <
- Replaces `>` 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 white space

AWS WAF Classic 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 Classic to search for in web requests. The maximum length is 50 bytes. If you're base64-encoding the value, the 50-byte maximum length 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/wafv2/.
2. In the navigation pane, choose **String and regex 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. 181).
   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/wafv2/.
2. Remove the string match condition from the rules that are using it:
   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, choose **Edit rule**.
d. Choose the **X** next to the condition you want to delete.
e. Choose **Update**.
f. Repeat for all the remaining rules that are using the string match condition that you want to delete.

3. Remove the filters from the condition you want to delete:
   a. In the navigation pane, choose **String and regex matching**.
   b. Choose the name of the string match condition that you want to delete.
   c. In the right pane, choose the check box next to **Filter** in order to select all of the filters.
   d. Choose the **Delete filter**.
4. In the navigation pane, choose **String and regex matching**.
5. In the **String and regex match conditions** pane, choose the string match condition that you want to delete.
6. Choose **Delete** to delete the selected condition.

Working with regex match conditions

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

If you want to allow or block web requests based on strings that match a regular expression (regex) pattern that appears in the requests, create one or more regex match conditions. A regex match condition is a type of string match condition that identifies the pattern 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 Classic to inspect for the pattern. Later in the process, when you create a web ACL, you specify whether to allow or block requests that contain the pattern.

Topics

Creating a regex match condition

When you create regex match conditions, you specify pattern sets that identify the string (using a regular expression) that you want to search for. You then add those pattern sets to filters that specify the part of web requests that you want AWS WAF Classic to inspect for that pattern set, such as the URI or the query string.

You can add multiple regular expressions to a single pattern set. If you do so, those expressions are combined with an *OR*. That is, a web request will match the pattern set if the appropriate part of the request matches any of the expressions listed.

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

AWS WAF Classic supports most standard Perl Compatible Regular Expressions (PCRE). However, the following are not supported:

- Backreferences and capturing subexpressions
- Arbitrary zero-width assertions
- Subroutine references and recursive patterns
- Conditional patterns
- Backtracking control verbs
- The \C single-byte directive
- The \R newline match directive
- The \K start of match reset directive
- Callouts and embedded code
- Atomic grouping and possessive quantifiers

To create a regex match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **String and regex matching**.
3. Choose **Create condition**.
4. Specify the applicable filter settings. For more information, see Values that you specify when you create or edit RegEx match conditions (p. 186).
5. Choose **Create pattern set and add filter** (if you created a new pattern set) or **Add filter** if you used an existing pattern set.
6. Choose **Create**.

Values that you specify when you create or edit RegEx match conditions

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

Name

Enter a name for the regex match condition. The name can contain only alphanumeric characters (A-Z, a-z, 0-9) or the following special characters: _-!"#`+*},./. You can't change the name of a condition after you create it.

Type

Choose **Regex match**.

Part of the request to filter on

Choose the part of each web request that you want AWS WAF Classic to inspect for the pattern 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 URI path of the request, which identifies the resource, for example, `/images/daily-ad.jpg`. This doesn't include the query string or fragment components of the URI. For information, see Uniform Resource Identifier (URI): Generic Syntax.

Unless a **Transformation** is specified, a URI is not normalized and is inspected just as AWS receives it from the client as part of the request. A **Transformation** will reformat the URI as specified.

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 Classic 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 Classic gets the length of the body from the request headers.) For more information, see Working with size constraint conditions (p. 171).

Single query parameter (value only)

Any parameter that you have defined as part of the query string. For example, if the URL is "www.xyz.com?UserName=abc&SalesRegion=seattle" you can add a filter to either the *UserName* or *SalesRegion* parameter.

If duplicate parameters appear in the query string, the values are evaluated as an "OR." That is, either value will trigger a match. For example, in the URL "www.xyz.com?SalesRegion=boston&SalesRegion=seattle", a pattern that matches either "boston" or "seattle" in **Value to match** will trigger a match.

If you choose **Single query parameter (value only)** you will also specify a **Query parameter name**. This is the parameter in the query string that you will inspect, such as *UserName* or *SalesRegion*. The maximum length for **Query parameter name** is 30 characters. **Query parameter name** is not case sensitive. For example, it you specify *UserName* as the **Query parameter name**, this will match all variations of *UserName*, such as *username* and *UsERName*.

All query parameters (values only)

Similar to **Single query parameter (value only)**, but rather than inspecting the value of a single parameter, AWS WAF Classic inspects the value of all parameters within the query string for the pattern specified in the **Value to match**. For example, in the URL "www.xyz.com?UserName=abc&SalesRegion=seattle", a pattern in **Value to match** that matches either the value in *UserName* or *SalesRegion* will trigger a match.

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 enter the name of a header that you want AWS WAF Classic to inspect.

Transformation

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

You can only specify a single type of text transformation.

Transformations can perform the following operations:

None

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

HTML decode

AWS WAF Classic replaces HTML-encoded characters with unencoded characters:

- Replaces `&quot;` with `&`
- Replaces ` ` with a non-breaking space
- Replaces `<` with <
- Replaces `>` 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 white space

AWS WAF Classic 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.

Regex pattern to match to request

You can choose an existing pattern set, or create a new one. If you create a new one specify the following:

New pattern set name

Enter a name and then specify the regex pattern that you want AWS WAF Classic to search for.

If you add multiple regular expressions to a pattern set, those expressions are combined with an *OR*. That is, a web request will match the pattern set if the appropriate part of the request matches any of the expressions listed.

The maximum length of **Value to match** is 70 characters.

Editing a regex match condition

You can make the following changes to an existing regex match condition:

- Delete a pattern from an existing pattern set
- Add a pattern to an existing pattern set
- Delete a filter to an existing regex match condition
- Add a filter to an existing regex match condition (You can have only one filter in a regex match condition. Therefore, in order to add a filter, you must delete the existing filter first.)
- Delete an existing regex match condition

> **Note**
> You cannot add or delete a pattern set from an existing filter. You must either edit the pattern set, or delete the filter and create a new filter with a new pattern set.

To delete a pattern from an existing pattern set

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **String and regex matching**.
3. Choose **View regex pattern sets**.
4. Choose the name of the pattern set you want to edit.
5. Choose **Edit**.
6. Choose the **X** next to the pattern you want to delete.
7. Choose **Save**.

To add a pattern to an existing pattern set

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **String and regex matching**.
3. Choose **View regex pattern sets**.
4. Choose the name of the pattern set to edit.
5. Choose **Edit**.
6. Enter a new regex pattern.
7. Choose the **+** next to the new pattern.
8. Choose **Save**.

To delete a filter from an existing regex match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **String and regex matching**.
3. Choose the name of the condition with the filter you want to delete.
4. Choose the box next to the filter you want to delete.

5. Choose **Delete filter**.

To delete a regex match condition

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. Delete the filter from the regex condition. See To delete a filter from an existing regex match condition (p. 189) for instructions to do this.)
3. Remove the regex match condition from the rules that are using it:
   a. In the navigation pane, choose **Rules**.
   b. Choose the name of a rule that is using the regex match condition that you want to delete.
   c. In the right pane, choose **Edit rule**.
   d. Choose the **X** next to the condition you want to delete.
   e. Choose **Update**.
   f. Repeat for all the remaining rules that are using the regex match condition that you want to delete.
4. In the navigation pane, choose **String and regex matching**.
5. Select the button next to the condition you want to delete.
6. Choose **Delete**.

To add or change a filter to an existing regex match condition

You can have only one filter in a regex match condition. If you want to add or change the filter, you must first delete the existing filter.

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. Delete the filter from the regex condition you want to change. See To delete a filter from an existing regex match condition (p. 189) for instructions to do this.)
3. In the navigation pane, choose **String and regex matching**.
4. Choose the name of the condition you want to change.
5. Choose **Add filter**.
6. Enter the appropriate values for the new filter and choose **Add**.

Working with rules

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

Rules let you precisely target the web requests that you want AWS WAF Classic to allow or block by specifying the exact conditions that you want AWS WAF Classic to watch for. For example, AWS WAF Classic 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

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

If you add more than one condition to a rule, a web request must match all the conditions for AWS WAF Classic 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/wafv2/.
2. In the navigation pane, choose **Rules**.
3. Choose **Create rule**.
4. Enter the following values:

   Name

   Enter a name.

   CloudWatch metric name

   Enter a name for the CloudWatch metric that AWS WAF Classic will create and will associate with the rule. The name can contain only alphanumeric characters (A-Z, a-z, 0-9), with maximum length 128 and minimum length one. It can't contain white space or metric names reserved for AWS WAF Classic, including "All" and "Default_Action.

   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 an IP address in a five-minute period. For more information about these rule types, see How AWS WAF Classic works (p. 141).

   Rate limit

   For a rate-based rule, enter the maximum number of requests to allow in any five-minute period from an IP address that matches the rule's conditions. The rate limit must be at least 100.

   You can specify a rate limit alone, or a rate limit and conditions. If you specify only a rate limit, AWS WAF places the limit on all IP addresses. If you specify a rate limit and conditions, AWS WAF places the limit on IP addresses that match the conditions.

   When an IP address reaches the rate limit threshold, AWS WAF applies the assigned action (block or count) as quickly as possible, usually within 30 seconds. Once the action is in place, if five minutes pass with no requests from the IP address, AWS WAF resets the counter to zero.
5. To add a condition to the rule, specify the following values:

   When a request does/does not

   If you want AWS WAF Classic to allow or block requests based on the filters in a condition, choose **does**. For example, if an IP match condition includes the IP address range 192.0.2.0/24

and you want AWS WAF Classic to allow or block requests that come from those IP addresses, choose **does**.

If you want AWS WAF Classic 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 Classic 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**
- Geo match conditions – choose **originate from a geographic location 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**
- Regular expression match conditions – choose **match at least one of the filters in the regex 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 Classic to allow or block requests based on that rule
   - If you add two IP match conditions to the same rule, AWS WAF Classic 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

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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/wafv2/.
2. In the navigation pane, choose **Rules**.
3. Choose the name of the rule in which you want to add or remove conditions.
4. Choose **Add rule**.

5. To add a condition, choose **Add condition** and specify the following values:

   When a request does/does not

   If you want AWS WAF Classic 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 Classic 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 Classic 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**
   - Geo match conditions – choose **originate from a geographic location 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**
   - Regular expression match conditions – choose **match at least one of the filters in the regex 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 remove a condition, select the **X** to the right of the condition name
7. Choose **Update**.

Deleting a rule

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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/wafv2/.
2. To remove the rule from the web ACLs that are using it, perform the following steps for each of the web ACLs:

   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. Choose the **Rules** tab.
d. Choose **Edit web ACL**.
e. Choose the **X** to the right of the rule that you want to delete, and then choose **Update**.

3. In the navigation pane, choose **Rules**.
4. Select the name of the rule you want to delete.
5. Choose **Delete**.

AWS Marketplace rule groups

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

AWS WAF Classic provides *AWS Marketplace rule groups* to help you protect your resources. AWS Marketplace rule groups are collections of predefined, ready-to-use rules that are written and updated by AWS and AWS partner companies.

Some AWS Marketplace rule groups are designed to help protect specific types of web applications like WordPress, Joomla, or PHP. Other AWS Marketplace rule groups offer broad protection against known threats or common web application vulnerabilities, such as those listed in the OWASP Top 10.

You can install a single AWS Marketplace rule group from your preferred AWS partner, and you can also add your own customized AWS WAF Classic rules for increased protection. If you are subject to regulatory compliance like PCI or HIPAA, you might be able to use AWS Marketplace rule groups to satisfy web application firewall requirements.

AWS Marketplace rule groups are available with no long-term contracts, and no minimum commitments. When you subscribe to a rule group, you are charged a monthly fee (prorated hourly) and ongoing request fees based on volume. For more information, see AWS WAF Classic Pricing and the description for each AWS Marketplace rule group on AWS Marketplace.

Automatic updates

Keeping up to date on the constantly changing threat landscape can be time consuming and expensive. AWS Marketplace rule groups can save you time when you implement and use AWS WAF Classic. Another benefit is that AWS and our AWS partners automatically update AWS Marketplace rule groups when new vulnerabilities and threats emerge.

Many of our partners are notified of new vulnerabilities before public disclosure. They can update their rule groups and deploy them to you even before a new threat is widely known. Many also have threat research teams to investigate and analyze the most recent threats in order to write the most relevant rules.

Access to the rules in an AWS Marketplace rule group

Each AWS Marketplace rule group provides a comprehensive description of the types of attacks and vulnerabilities that it's designed to protect against. To protect the intellectual property of the rule group providers, you can't view the individual rules within a rule group. This restriction also helps to keep malicious users from designing threats that specifically circumvent published rules.

Because you can’t view individual rules in an AWS Marketplace rule group, you also can't edit any rules in an AWS Marketplace rule group. However, you can exclude specific rules from a rule group. This is called a "rule group exception." Excluding rules does not remove those rules. Rather, it changes the action for

the rules to `COUNT`. Therefore, requests that match an excluded rule are counted but not blocked. You will receive COUNT metrics for each excluded rule.

Excluding rules can be helpful when troubleshooting rule groups that are blocking traffic unexpectedly (false positives). One troubleshooting technique is to identify the specific rule within the rule group that is blocking the desired traffic and then disable (exclude) that particular rule.

In addition to excluding specific rules, you can refine your protection by enabling or disabling entire rule groups, as well as choosing the rule group action to perform. For more information, see Using AWS Marketplace rule groups (p. 195).

Quotas

You can enable only one AWS Marketplace rule group. You can also enable one custom rule group that you create using AWS Firewall Manager. These rule groups count towards the 10 rule maximum quota per web ACL. Therefore, you can have one AWS Marketplace rule group, one custom rule group, and up to eight custom rules in a single web ACL.

Pricing

For AWS Marketplace rule group pricing, see AWS WAF Classic Pricing and the description for each AWS Marketplace rule group on AWS Marketplace.

Using AWS Marketplace rule groups

You can subscribe to and unsubscribe from AWS Marketplace rule groups on the AWS WAF Classic console. You can also exclude specific rules from a rule group.

To subscribe to and use an AWS Marketplace rule group

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, choose **Marketplace**.
3. In the **Available marketplace products** section, choose the name of a rule group to view the details and pricing information.
4. If you want to subscribe to the rule group, choose **Continue**.

   Note
   If you don't want to subscribe to this rule group, simply close this page in your browser.
5. Choose **Set up your account**.
6. Add the rule group to a web ACL, just as you would add an individual rule. For more information, see Creating a Web ACL (p. 198) or Editing a Web ACL (p. 202).

   Note
   When adding a rule group to a web ACL, the action that you set for the rule group (either **No override** or **Override to count**) is called the rule group override action. For more information, see Rule group override (p. 196).

To unsubscribe from an AWS Marketplace rule group

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. Remove the rule group from all web ACLs. For more information, see Editing a Web ACL (p. 202).
3. In the navigation pane, choose **Marketplace**.
4. Choose **Manage your subscriptions**.

5. Choose **Cancel subscription** next to the name of the rule group that you want to unsubscribe from.
6. Choose **Yes, cancel subscription**.

To exclude a rule from a rule group (rule group exception)

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
2. If not already enabled, enable AWS WAF Classic logging. For more information, see Logging Web ACL traffic information (p. 213). Use the AWS WAF Classic logs to identify the IDs of the rules that you want to exclude. These are typically rules that are blocking legitimate requests.
3. In the navigation pane, choose **Web ACLs**.
4. Choose the web ACL that you want to edit.

   Note
   The rule group that you want to edit must be associated with a web ACL before you can exclude a rule from that rule group.
5. On the **Rules** tab in the right pane, choose **Edit web ACL**.
6. In the **Rule group exceptions** section, expand the rule group that you want to edit.
7. Choose the **X** next to the rule that you want to exclude. You can identify the correct rule ID by using the AWS WAF Classic logs.
8. Choose **Update**.

Excluding rules does not remove those rules from the rule group. Rather, it changes the action for the rules to `COUNT`. Therefore, requests that match an excluded rule are counted but not blocked. You will receive `COUNT` metrics for each excluded rule.

Note
You can use this same procedure to exclude rules from custom rule groups that you have created in AWS Firewall Manager. However, rather than excluding a rule from a custom rule group using these steps, you can also simply edit a custom rule group using the steps described in Adding and deleting rules from an AWS WAF Classic rule group (p. 207).

Rule group override

AWS Marketplace rule groups have two possible actions: **No override** and **Override to count**. If you want to test the rule group, set the action to **Override to count**. This rule group action overrides any *block* action that is specified by individual rules contained within the group. That is, if the rule group's action is set to **Override to count**, instead of potentially blocking matching requests based on the action of individual rules within the group, those requests will be counted. Conversely, if you set the rule group's action to **No override**, actions of the individual rules within the group will be used.

Troubleshooting AWS Marketplace rule groups

If you find that an AWS Marketplace rule group is blocking legitimate traffic, perform the following steps.

To troubleshoot an AWS Marketplace rule group

1. Exclude the specific rules that are blocking legitimate traffic. You can identify which rules are blocking which requests using the AWS WAF Classic logs. For more information about excluding rules, see To exclude a rule from a rule group (rule group exception) (p. 196).
2. If excluding specific rules does not solve the problem, you can change the action for the AWS Marketplace rule group from **No override** to **Override to count**. This allows the web request to pass through, regardless of the individual rule actions within the rule group. This also provides you with Amazon CloudWatch metrics for the rule group.

3. After setting the AWS Marketplace rule group action to **Override to count**, contact the rule group provider's customer support team to further troubleshoot the issue. For contact information, see the rule group listing on the product listing pages on AWS Marketplace.

Contacting customer support

For problems with AWS WAF Classic or a rule group that is managed by AWS, contact AWS Support. For problems with a rule group that is managed by an AWS partner, contact that partner's customer support team. To find partner contact information, see the partner's listing on AWS Marketplace.

Creating and selling AWS Marketplace rule groups

If you want to sell AWS Marketplace rule groups on AWS Marketplace, see How to Sell Your Software on AWS Marketplace.

Working with web ACLs

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

When you add rules to a web ACL, you specify whether you want AWS WAF Classic 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 Classic 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 Classic 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 Classic 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. 197).

If you want to test a rule before you start using it to allow or block requests, you can configure AWS WAF Classic to count the web requests that match the conditions in the rule. For more information, see Testing web ACLs (p. 203).

Topics

Deciding on the default action for a Web ACL

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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 Classic to allow web requests or to block web requests. The default action indicates what you want AWS WAF Classic 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 originate from countries that either you don't do business in or are the frequent source of attacks
- Requests that include fake values in the **User-Agent** header
- Requests that appear to include malicious SQL code

Creating a Web ACL

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

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/wafv2/.
2. If this is your first time using AWS WAF Classic, choose **Go to AWS WAF Classic** and then **Configure Web ACL**. If you've used AWS WAF Classic before, choose **Web ACLs** in the navigation pane, and then choose **Create web ACL**.
3. For **Web ACL name**, enter 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), with maximum length 128 and minimum length one. It can't contain white space or metric names reserved for AWS WAF Classic, including "All" and "Default_Action."

   > **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 Classic 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. 162)
   - Working with IP match conditions (p. 167)
   - Working with geographic match conditions (p. 169)
   - Working with size constraint conditions (p. 171)
   - Working with SQL injection match conditions (p. 175)
   - Working with string match conditions (p. 180)
   - Working with regex match conditions (p. 185)

8. If you've already created the rules or rule groups (or subscribed to an AWS Marketplace rule group) 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. Enter the following values:

      Name

      Enter a name.

      CloudWatch metric name

      Enter a name for the CloudWatch metric that AWS WAF Classic will create and will associate with the rule. The name can contain only alphanumeric characters (A-Z, a-z, 0-9), with maximum length 128 and minimum length one. It can't contain white space or metric names reserved for AWS WAF Classic, including "All" and "Default_Action."

      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 Classic 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 Classic 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 Classic 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**
      - Geo match conditions – choose **originate from a geographic location 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**
- Regex match conditions – choose **match at least one of the filters in the regex 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 Classic to allow or block requests based on that rule.
- If you add two IP match conditions to the same rule, AWS WAF Classic 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 or rule group in the web ACL, choose the kind of management you want AWS WAF Classic to provide, as follows:

- For each rule, choose whether you want AWS WAF Classic to allow, block, or count web requests based on the conditions in the rule:
  - **Allow** – API Gateway, 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** – API Gateway, 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 Classic with CloudFront custom error pages (p. 219).
  - **Count** – AWS WAF Classic 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. 203).
- For each rule group, set the override action for the rule group:
  - **No override** – Causes the actions of the individual rules within the rule group to be used.
  - **Override to count** – Overrides any block actions that are specifieid by individual rules in the group, so that all matching requests are only counted.

  For more information, see Rule group override (p. 196).

11. If you want to change the order of the rules in the web ACL, use the arrows in the **Order** column. AWS WAF Classic 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 Classic 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. 197).

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 an Amazon API Gateway API, a CloudFront distribution or an Application Load Balancer

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

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 Classic to Control Access to Your Content in the *Amazon CloudFront Developer Guide*.

The following restrictions apply when associating a web ACL:

- Each API Gateway API, Application Load Balancer and CloudFront distribution can be associated with only one web ACL.
- Web ACLs associated with a CloudFront distribution cannot be associated with an Application Load Balancer or API Gateway API. The web ACL can, however, be associated with other CloudFront distributions.

To associate a web ACL with an API Gateway API, 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/wafv2/.
2. In the navigation pane, choose **Web ACLs**.
3. Choose the web ACL that you want to associate with an API Gateway API, 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 API Gateway API, 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 an additional API Gateway API, CloudFront distribution or another Application Load Balancer, repeat steps 4 through 6.

To disassociate a web ACL from an API Gateway API, 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/wafv2/.
2. In the navigation pane, choose **Web ACLs**.
3. Choose the web ACL that you want to disassociate from an API Gateway API, CloudFront distribution or Application Load Balancer.

4. On the **Rules** tab, under **AWS resources using this web ACL**, choose the **x** for each API Gateway API, CloudFront distribution or Application Load Balancer that you want to disassociate this web ACL from.

Editing a Web ACL

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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/wafv2/.
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 Classic 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 Classic, 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.

   Note
   When setting the action for a rule group or an AWS Marketplace rule group (as opposed to a single rule), the action you set for the rule group (either **No override** or **Override to count**) is called the override action. For more information, see Rule group override (p. 196)
9. Choose **Save changes**.

Deleting a Web ACL

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

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/wafv2/.
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 Classic, 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 API Gateway API, 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

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

To ensure that you don't accidentally configure AWS WAF Classic 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 Classic 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 Classic 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 Classic increments a counter for that rule. Then AWS WAF Classic inspects the web request based on the conditions in the next rule. If the request matches all the conditions in that rule, AWS WAF Classic increments a counter for the rule. This continues until AWS WAF Classic 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 an Amazon API Gateway API, 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 API Gateway, CloudFront or an Application Load Balancer forwards to AWS WAF Classic 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 Classic with CloudFront is a global service and metrics are available only when you choose the **US East (N. Virginia)** Region in the AWS Management Console. If you choose another region, no AWS WAF Classic 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 an API Gateway API, 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 API Gateway API, 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 API Gateway CloudFront or an Application Load Balancer has forwarded to AWS WAF Classic. For more information, see Viewing a sample of the web requests that API Gateway CloudFront or an Application Load Balancer has forwarded to AWS WAF Classic (p. 205).
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. 160).

   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. 202).

Viewing a sample of the web requests that API Gateway CloudFront or an Application Load Balancer has forwarded to AWS WAF Classic

In the AWS WAF Classic console, you can view a sample of the requests that API Gateway CloudFront or an Application Load Balancer has forwarded to AWS WAF Classic 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 API Gateway APIs, 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 API Gateway; CloudFront or an Application Load Balancer has forwarded to AWS WAF Classic

1. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
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 URI path of the request, which identifies the resource, for example, `/images/daily-ad.jpg`. This doesn't include the query string or fragment components of the URI. For information, see Uniform Resource Identifier (URI): Generic Syntax.

   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 Classic 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 Classic received the request from API Gateway, 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 Classic 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**.

Working with AWS WAF Classic rule groups for use with AWS Firewall Manager

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

An AWS WAF Classic *rule group* is a set of rules that you add to an AWS WAF Classic AWS Firewall Manager policy. You can create your own rule group, or you can purchase a managed rule group from AWS Marketplace.

Important
If you want to add an AWS Marketplace rule group to your Firewall Manager policy, each account in your organization must first subscribe to that rule group. After all accounts have subscribed, you can then add the rule group to a policy. For more information, see AWS Marketplace rule groups (p. 194).

Topics

Creating an AWS WAF Classic rule group

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).

For the latest version of AWS WAF, see AWS WAF (p. 6).

When you create an AWS WAF Classic rule group to use with AWS Firewall Manager, you specify which rules to add to the group.

To create a rule group (console)

1. Sign in to the AWS Management Console using the AWS Firewall Manager administrator account that you set up in the prerequisites, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fms.

   Note
   For information about setting up a Firewall Manager administrator account, see Step 2: Set the AWS Firewall Manager administrator account (p. 250).
2. In the navigation pane, choose **Switch to AWS WAF Classic**.
3. In the AWS WAF Classic navigation pane, choose **Rule groups**.
4. Choose **Create rule group**.

   Note
   You can't add rate-based rules to a rule group.
5. If you have already created the rules that you want to add to the rule group, choose **Use existing rules for this rule group** . If you want to create new rules to add to the rule group, choose **Create rules and conditions for this rule group**.
6. Choose **Next**.
7. If you chose to create rules, follow the steps to create them at Creating a rule and adding conditions (p. 191).

   Note
   Use the AWS WAF Classic console to create your rules.

   When you've created all the rules you need, go to the next step.
8. Type a rule group name.
9. To add a rule to the rule group, select a rule then choose **Add rule**. Choose whether to allow, block, or count requests that match the rule's conditions. For more information on the choices, see How AWS WAF Classic works (p. 141).
10. When you are finished adding rules, choose **Create**.

You can test your rule group by adding it to an AWS WAF WebACL and setting the WebACL action to **Override to Count**. This action overrides any action that you choose for the rules contained in the group, and only counts matching requests. For more information, see Creating a Web ACL (p. 198).

Adding and deleting rules from an AWS WAF Classic rule group

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

You can add or delete rules in an AWS WAF Classic rule group.

Deleting a rule from the rule group does not delete the rule itself. It only removes the rule from the rule group.

To add or delete rules in a rule group (console)

1. Sign in to the AWS Management Console using the AWS Firewall Manager administrator account that you set up in the prerequisites, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fms.

   Note
   For information about setting up a Firewall Manager administrator account, see Step 2: Set the AWS Firewall Manager administrator account (p. 250).
2. In the navigation pane, choose **Switch to AWS WAF Classic**.
3. In the AWS WAF Classic navigation pane, choose **Rule groups**.
4. Choose the rule group that you want to edit.
5. Choose **Edit rule group**.
6. To add rules, perform the following steps:

   a. Select a rule, and then choose **Add rule to rule group**. Choose whether to allow, block, or count requests that match the rule's conditions. For more information on the choices, see How AWS WAF Classic works (p. 141). Repeat to add more rules to the rule group.

      Note
      You cannot add rate-based rules to rule group.

   b. Choose **Update**.
7. To delete rules, perform the following steps:

   a. Choose the **X** next to the rule to delete. Repeat to delete more rules from the rule group.
   b. Choose **Update**.

Getting started with AWS Firewall Manager to enable AWS WAF Classic rules

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

You can use AWS Firewall Manager to enable AWS WAF rules, AWS WAF Classic rules, AWS Shield Advanced protections, and Amazon VPC security groups. The steps for getting set up are slightly different for each:

- To use Firewall Manager to enable rules using the latest version of AWS WAF, don't use this topic. Instead, follow the steps in Getting started with AWS Firewall Manager AWS WAF policies (p. 254).
- To use Firewall Manager to enable AWS Shield Advanced protections, follow the steps in Getting started with AWS Firewall Manager AWS Shield Advanced policies (p. 256).
- To use Firewall Manager to enable Amazon VPC security groups, follow the steps in Getting started with AWS Firewall Manager Amazon VPC security group policies (p. 259).

To use Firewall Manager to enable AWS WAF Classic rules, perform the following steps in sequence.

Topics

Step 1: Complete the prerequisites

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

There are several mandatory steps to prepare your account for AWS Firewall Manager. Those steps are described in AWS Firewall Manager prerequisites (p. 249). Complete all the prerequisites before proceeding to Step 2: Create rules (p. 209).

Step 2: Create rules

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

In this step, you create rules using AWS WAF Classic. If you already have AWS WAF Classic rules that you want to use with AWS Firewall Manager, skip this step and go to Step 3: Create a rule group (p. 209).

Note
Use the AWS WAF Classic console to create your rules.

To create AWS WAF Classic rules (console)

- Create your rules, and then add your conditions to your rules. For more information, see Creating a rule and adding conditions (p. 191).

You are now ready to go to Step 3: Create a rule group (p. 209).

Step 3: Create a rule group

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

A rule group is a set of rules that defines what actions to take when a particular set of conditions is met. You can use managed rule groups from AWS Marketplace, and you can create your own rule groups. For information about managed rule groups, see AWS Marketplace rule groups (p. 194).

To create your own rule group, perform the following procedure.

To create a rule group (console)

1. Sign in to the AWS Management Console using the AWS Firewall Manager administrator account that you set up in the prerequisites, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fms.

2. In the navigation pane, choose **Security policies**.
3. If you have not met the prerequisites, the console displays instructions about how to fix any issues. Follow the instructions, and then begin this step (create a rule group) again. If you have met the prerequisites, choose **Close**.
4. Choose **Create policy**.

   For **Policy type**, choose **AWS WAF Classic**.
5. Choose **Create an AWS Firewall Manager policy and add a new rule group**.
6. Choose an AWS Region, and then choose **Next**.
7. Because you already created rules, you don't need to create conditions. Choose **Next**.
8. Because you already created rules, you don't need to create rules. Choose **Next**.
9. Choose **Create rule group**.
10. For **Name**, enter a friendly name.
11. Enter a name for the CloudWatch metric that AWS WAF Classic will create and will associate with the rule group. The name can contain only alphanumeric characters (A-Z, a-z, 0-9) or the following special characters: _-!"#`+*},./. It can't contain white space.
12. Select a rule, and then choose **Add rule**. A rule has an action setting that allows you to choose whether to allow, block, or count requests that match the rule's conditions. For this tutorial, choose **Count**. Repeat adding rules until you have added all the rules that you want to the rule group.
13. Choose **Create**.

You are now ready to go to Step 4: Create and apply an AWS Firewall Manager AWS WAF Classic policy (p. 210).

Step 4: Create and apply an AWS Firewall Manager AWS WAF Classic policy

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

After you create the rule group, you create an AWS Firewall Manager AWS WAF policy. A Firewall Manager AWS WAF policy contains the rule group that you want to apply to your resources.

To create a Firewall Manager AWS WAF policy (console)

1. After you create the rule group (the last step in the preceding procedure, Step 3: Create a rule group (p. 209)), the console displays the **Rule group summary** page. Choose **Next**.
2. For **Name**, enter a friendly name.
3. For **Policy type**, choose **WAF**.
4. For **Region**, choose an AWS Region. To protect Amazon CloudFront resources, choose **Global**.

   To protect resources in multiple regions (other than CloudFront resources), you must create separate Firewall Manager policies for each Region.
5. Select a rule group to add, and then choose **Add rule group**.
6. A policy has two possible actions: **Action set by rule group** and **Count**. If you want to test the policy and rule group, set the action to **Count**. This action overrides any *block* action specified by the rule group contained in the policy. That is, if the policy's action is set to **Count**, those requests are only

counted and not blocked. Conversely, if you set the policy's action to **Action set by rule group**, actions of the rule group in the policy are used. For this tutorial, choose **Count**.

7. Choose **Next**.
8. If you want to include only specific accounts in the policy, or alternatively exclude specific accounts from the policy, select **Select accounts to include/exclude from this policy (optional)**. Choose either **Include only these accounts in this policy** or **Exclude these accounts from this policy**. You can choose only one option. Choose **Add**. Select the account numbers to include or exclude, and then choose **OK**.

   Note
   If you don't select this option, Firewall Manager applies a policy to all accounts in your organization in AWS Organizations. If you add a new account to the organization, Firewall Manager automatically applies the policy to that account.
9. Choose the types of resources that you want to protect.
10. If you want to protect only resources with specific tags, or alternatively exclude resources with specific tags, select **Use tags to include/exclude resources**, enter the tags, and then choose either **Include** or **Exclude**. You can choose only one option.

    If you enter more than one tag (separated by commas), and if a resource has any of those tags, it is considered a match.

    For more information about tags, see Working with Tag Editor.
11. Choose **Create and apply this policy to existing and new resources**.

    This option creates a web ACL in each applicable account within an organization in AWS Organizations, and associates the web ACL with the specified resources in the accounts. This option also applies the policy to all new resources that match the preceding criteria (resource type and tags). Alternatively, if you choose **Create but do not apply this policy to existing or new resources**, Firewall Manager creates a web ACL in each applicable account within the organization, but doesn't apply the web ACL to any resources. You must apply the policy to resources later.
12. Leave the choice for **Replace existing associated web ACLs** at the default setting.

    When this option is selected, Firewall Manager removed all existing web ACL associations from in-scope resources before it associates the new policy's web ACLs to them.
13. Choose **Next**.
14. Review the new policy. To make any changes, choose **Edit**. When you are satisfied with the policy, choose **Create policy**.

Tutorial: Creating a AWS Firewall Manager policy with hierarchical rules

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

With AWS Firewall Manager, you can create and apply AWS WAF Classic protection policies that contain hierarchical rules. That is, you can create and enforce certain rules centrally, but delegate the creation and maintenance of account-specific rules to other individuals. You can monitor the centrally applied (common) rules for any accidental removal or mishandling, thereby ensuring that they are applied consistently. The account-specific rules add further protection customized for the needs of individual teams.

Note
In the latest version of AWS WAF, this capability is built in and doesn't require any special handling. If you aren't already using AWS WAF Classic, use the latest version instead. See Creating an AWS Firewall Manager policy for AWS WAF (p. 266).

The following tutorial describes how to create a hierarchical set of protection rules.

Topics

Step 1: Designate a Firewall Manager administrator account

To use AWS Firewall Manager, you must designate an account in your organization as the Firewall Manager administrator account. This account can be either the management account or a member account in the organization.

You can use the Firewall Manager administrator account to create a set of common rules that you apply to other accounts in the organization. Other accounts in the organization can't change these centrally applied rules.

To designate an account as a Firewall Manager administrator account and complete other prerequisites for using Firewall Manager, see the instructions in AWS Firewall Manager prerequisites (p. 249). If you've already completed the prerequisites, you can skip to step 2 of this tutorial.

In this tutorial, we refer to the administrator account as **`Firewall-Administrator-Account`**.

Step 2: Create a rule group using the Firewall Manager administrator account

Next, create a rule group using **`Firewall-Administrator-Account`**. This rule group contains the common rules that you will apply to all member accounts governed by the policy that you create in the next step. Only **`Firewall-Administrator-Account`** can make changes to these rules and the container rule group.

In this tutorial, we refer to this container rule group as **`Common-Rule-Group`**.

To create a rule group, see the instructions in Creating an AWS WAF Classic rule group (p. 206). Remember to sign in to the console using your Firewall Manager administrator account (**`Firewall-Administrator-Account`**) when following these instructions.

Step 3: Create a Firewall Manager policy and attach the common rule group

Using **`Firewall-Administrator-Account`**, create a Firewall Manager policy. When you create this policy, you must do the following:

- Add **`Common-Rule-Group`** to the new policy.

- Include all accounts in the organization that you want **Common-Rule-Group** applied to.
- Add all resources that you want **Common-Rule-Group** applied to.

For instructions on creating a policy, see Creating an AWS Firewall Manager policy (p. 266).

This creates a web ACL in each specified account and adds **Common-Rule-Group** to each of those web ACLs. After you create the policy, this web ACL and the common rules are deployed to all specified accounts.

In this tutorial, we refer to this web ACL as **Administrator-Created-ACL**. A unique **Administrator-Created-ACL** now exists in each specified member account of the organization.

Step 4: Add account-specific rules

Each member account in the organization can now add their own account-specific rules to the **Administrator-Created-ACL** that exists in their account. The common rules already in **Administrator-Created-ACL** continue to apply, along with the new, account-specific rules. AWS WAF inspects web requests based on the order in which rules appear in the web ACL. This applies to both **Administrator-Created-ACL** and account-specific rules.

To add rules to **Administrator-Created-ACL**, see Editing a web ACL (p. 24).

Conclusion

You now have a web ACL that contains common rules administered by the Firewall Manager administrator account as well as account-specific rules maintained by each member account.

The **Administrator-Created-ACL** in each account references the single **Common-Rule-Group**. Therefore, future changes by the Firewall Manager administrator account to **Common-Rule-Group** will immediately take effect in each member account.

Member accounts can't change or remove the common rules in **Common-Rule-Group**.

Account-specific rules don't affect other accounts.

Logging Web ACL traffic information

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

You can enable logging to get detailed information about traffic that is analyzed by your web ACL. Information that is contained in the logs include the time that AWS WAF Classic received the request from your AWS resource, detailed information about the request, and the action for the rule that each request matched.

To get started, you set up an Amazon Kinesis Data Firehose. As part of that process, you choose a destination for storing your logs. Next, you choose the web ACL that you want to enable logging for. After you enable logging, AWS WAF delivers logs through the firehose to your storage destination.

For information about how to create an Amazon Kinesis Data Firehose and review your stored logs, see What Is Amazon Kinesis Data Firehose? To understand the permissions required for your Kinesis Data Firehose configuration, see Controlling Access with Amazon Kinesis Data Firehose.

You must have the following permissions to successfully enable logging:

- `iam:CreateServiceLinkedRole`
- `firehose:ListDeliveryStreams`
- `waf:PutLoggingConfiguration`

For more information about service-linked roles and the `iam:CreateServiceLinkedRole` permission, see Using service-linked roles for AWS WAF Classic (p. 239).

To enable logging for a web ACL

1. Create an Amazon Kinesis Data Firehose using a name starting with the prefix "aws-waf-logs-" For example, `aws-waf-logs-us-east-2-analytics`. Create the data firehose with a `PUT` source and in the region that you are operating. If you are capturing logs for Amazon CloudFront, create the firehose in US East (N. Virginia). For more information, see Creating an Amazon Kinesis Data Firehose Delivery Stream.

   Important
   Do not choose `Kinesis stream` as your source.
   One AWS WAF Classic log is equivalent to one Kinesis Data Firehose record. If you typically receive 10,000 requests per second and you enable full logs, you should have a 10,000 records per second setting in Kinesis Data Firehose. If you don't configure Kinesis Data Firehose correctly, AWS WAF Classic won't record all logs. For more information, see Amazon Kinesis Data Firehose Quotas.
2. Sign in to the AWS Management Console and open the AWS WAF console at https://console.aws.amazon.com/wafv2/.
3. In the navigation pane, choose **Web ACLs**.
4. Choose the web ACL that you want to enable logging for.
5. On the **Logging** tab, choose **Enable logging**.
6. Choose the Kinesis Data Firehose that you created in the first step. You must choose a firehose that begins with "aws-waf-logs-."
7. (Optional) If you don't want certain fields and their values included in the logs, redact those fields. Choose the field to redact, and then choose **Add**. Repeat as necessary to redact additional fields. The redacted fields appear as `xxx` in the logs. For example, if you redact the **cookie** field, the **cookie** field in the logs will be `xxx`.
8. Choose **Enable logging**.

   Note
   When you successfully enable logging, AWS WAF Classic will create a service linked role with the necessary permissions to write logs to the Amazon Kinesis Data Firehose. For more information, see Using service-linked roles for AWS WAF Classic (p. 239).

To disable logging for a web ACL

1. In the navigation pane, choose **Web ACLs**.
2. Choose the web ACL that you want to disable logging for.
3. On the **Logging** tab, choose **Disable logging**.
4. In the dialog box, choose **Disable logging**.

Example Example log

```
{
```

```
 "timestamp":1533689070589,
 "formatVersion":1,
 "webaclId":"385cb038-3a6f-4f2f-ac64-09ab912af590",
 "terminatingRuleId":"Default_Action",
 "terminatingRuleType":"REGULAR",
 "action":"ALLOW",
 "httpSourceName":"CF",
 "httpSourceId":"i-123",
 "ruleGroupList":[
                         {
                          "ruleGroupId":"41f4eb08-4e1b-2985-92b5-e8abf434fad3",
                          "terminatingRule":null,
                          "nonTerminatingMatchingRules":[
                                                          {"action" : "COUNT",
                                                          "ruleId" : "4659b169-2083-4a91-
bbd4-08851a9aaf74"}
                                                         ]
                          "excludedRules":               [
                                                          {"exclusionType" :
 "EXCLUDED_AS_COUNT",
                                                            "ruleId" : "5432a230-0113-5b83-
bbb2-89375c5bfa98"}
                                                         ]
                         }
                        ],

 "rateBasedRuleList":[
                             {
                              "rateBasedRuleId":"7c968ef6-32ec-4fee-96cc-51198e412e7f",
                              "limitKey":"IP",
                              "maxRateAllowed":100

                             },
                             {
                              "rateBasedRuleId":"462b169-2083-4a93-bbd4-08851a9aaf30",
                              "limitKey":"IP",
                              "maxRateAllowed":100
                              }
                              ],

 "nonTerminatingMatchingRules":[
                                        {"action" : "COUNT",

                                        "ruleId" : "4659b181-2011-4a91-bbd4-08851a9aaf52"}

                                       ],

 "httpRequest":{
                       "clientIp":"192.10.23.23",

                       "country":"US",

                       "headers":[

                                  {
                                   "name":"Host",
                                   "value":"127.0.0.1:1989"
                                  },
                                  {
                                   "name":"User-Agent",
                                   "value":"curl/7.51.2"
                                  },
                                  {
                                   "name":"Accept",
                                   "value":"*/*"
                                  }
```

```
                        ],
                    "uri":"REDACTED",
                    "args":"usernam=abc",
                    "httpVersion":"HTTP/1.1",
                    "httpMethod":"GET",
                    "requestId":"cloud front Request id"
                    }
}
```

Following is an explanation of each item listed in these logs:

timestamp

The timestamp in milliseconds.

formatVersion

The format version for the log.

webaclId

The GUID of the web ACL.

terminatingRuleId

The ID of the rule that terminated the request. If nothing terminates the request, the value is `Default_Action`.

terminatingRuleType

The type of rule that terminated the request. Possible values: RATE_BASED, REGULAR, and GROUP.

action

The action. Possible values for a terminating rule: ALLOW and BLOCK. COUNT is not a valid value for a terminating rule.

terminatingRuleMatchDetails

Detailed information about the terminating rule that matched the request. A terminating rule has an action that ends the inspection process against a web request. Possible actions for a terminating rule are ALLOW and BLOCK. This is only populated for SQL injection and cross-site scripting (XSS) match rule statements. As with all rule statements that inspect for more than one thing, AWS WAF applies the action on the first match and stops inspecting the web request. A web request with a terminating action could contain other threats, in addition to the one reported in the log.

httpSourceName

The source of the request. Possible values: CF (if the source is Amazon CloudFront), APIGW (if the source is Amazon API Gateway), and ALB (if the source is an Application Load Balancer).

httpSourceId

The source ID. This field shows the ID of the associated Amazon CloudFront distribution, the REST API for API Gateway, or the name for an Application Load Balancer.

ruleGroupList

The list of rule groups that acted on this request. In the preceding code example, there is only one.

ruleGroupId

The ID of the rule group. If the rule blocked the request, the ID for `ruleGroupID` is the same as the ID for `terminatingRuleId`.

terminatingRule

The rule within the rule group that terminated the request. If this is a non-null value, it also contains a **ruleid** and **action**. In this case, the action is always BLOCK.

nonTerminatingMatchingRules

The list of rules in the rule group that match the request. These are always COUNT rules (non-terminating rules that match).

action (nonTerminatingMatchingRules group)

This is always COUNT (non-terminating rules that match).

ruleId (nonTerminatingMatchingRules group)

The ID of the rule within the rule group that matches the request and was non-terminating. That is, COUNT rules.

excludedRules

The list of rules in the rule group that you have excluded. The action for these rules is set to COUNT.

exclusionType (excludedRules group)

A type that indicates that the excluded rule has the action COUNT.

ruleId (excludedRules group)

The ID of the rule within the rule group that is excluded.

rateBasedRuleList

The list of rate-based rules that acted on the request.

rateBasedRuleId

The ID of the rate-based rule that acted on the request. If this has terminated the request, the ID for `rateBasedRuleId` is the same as the ID for `terminatingRuleId`.

limitKey

The field that AWS WAF uses to determine if requests are likely arriving from a single source and thus subject to rate monitoring. Possible value: IP.

maxRateAllowed

The maximum number of requests, which have an identical value in the field that is specified by `limitKey`, allowed in a five-minute period. If the number of requests exceeds the `maxRateAllowed` and the other predicates specified in the rule are also met, AWS WAF triggers the action that is specified for this rule.

httpRequest

The metadata about the request.

clientIp

The IP address of the client sending the request.

country

The source country of the request. If AWS WAF is unable to determine the country of origin, it sets this field to –.

headers

The list of headers.

uri

The URI of the request. The preceding code example demonstrates what the value would be if this field had been redacted.

args

The query string.

httpVersion

The HTTP version.

httpMethod

The HTTP method in the request.

requestId

The ID of the request.

Listing IP addresses blocked by rate-based rules

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

AWS WAF Classic 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/wafv2/.
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 Classic works with Amazon CloudFront features

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

When you create a web ACL, you can specify one or more CloudFront distributions that you want AWS WAF Classic to inspect. AWS WAF Classic 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 Classic functionality. This chapter describes a few ways that you can configure CloudFront to make CloudFront and AWS WAF Classic work better together.

Topics

Using AWS WAF Classic with CloudFront custom error pages

When AWS WAF Classic 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 Classic 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 Classic 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 Classic. 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 Classic.

> **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 API Gateway CloudFront or an Application Load Balancer has forwarded to AWS WAF Classic (p. 205).

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

Using AWS WAF Classic with CloudFront for applications running on your own HTTP server

When you use AWS WAF Classic with CloudFront, you can protect your applications running on any HTTP webserver, whether it's a webserver that's running in Amazon Elastic Compute Cloud (Amazon EC2) or a webserver that you manage privately. You can also configure CloudFront to require HTTPS between CloudFront and your own webserver, as well as between viewers and CloudFront.

Requiring HTTPS Between CloudFront and Your Own Webserver

To require HTTPS between CloudFront and your own webserver, you can use the CloudFront custom origin feature and configure the **Origin Protocol Policy** and the **Origin Domain Name** settings for specific origins. In your CloudFront configuration, you can specify the DNS name of the server along with the port and the protocol that you want CloudFront to use when fetching objects from your origin. You should also ensure that the SSL/TLS certificate on your custom origin server matches the origin domain name you've configured. When you use your own HTTP webserver outside of AWS, you must use a certificate that is signed by a trusted third-party certificate authority (CA), for example, Comodo, DigiCert, or Symantec. For more information about requiring HTTPS for communication between CloudFront and your own webserver, see the topic Requiring HTTPS for Communication Between CloudFront and Your Custom Origin in the *Amazon CloudFront Developer Guide*.

Requiring HTTPS Between a Viewer and CloudFront

To require HTTPS between viewers and CloudFront, you can change the **Viewer Protocol Policy** for one or more cache behaviors in your CloudFront distribution. For more information about using HTTPS between viewers and CloudFront, see the topic Requiring HTTPS for Communication Between Viewers and CloudFront in the *Amazon CloudFront Developer Guide*. You can also bring your own SSL certificate so viewers can connect to your CloudFront distribution over HTTPS using your own domain name, for example *https://www.mysite.com*. For more information, see the topic Configuring Alternate Domain Names and HTTPS 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 Classic string match conditions to allow or block requests based on the HTTP method, as described in Working with string match conditions (p. 180). 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 Classic 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 Classic 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*.

Security in AWS WAF Classic

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).

For the latest version of AWS WAF, see AWS WAF (p. 6).

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. The effectiveness of our security is regularly tested and verified by third-party auditors as part of the AWS compliance programs. To learn about the compliance programs that apply to AWS WAF Classic, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your organization's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using AWS WAF Classic. The following topics show you how to configure AWS WAF Classic to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your AWS WAF Classic resources.

Topics

Data protection in AWS WAF Classic

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

The AWS shared responsibility model applies to data protection in AWS WAF Classic. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. This content includes the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.
- Set up API and user activity logging with AWS CloudTrail.

- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form fields such as a **Name** field. This includes when you work with AWS WAF Classic or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

AWS WAF Classic entities—such as web ACLs, rules, and conditions—are encrypted at rest, except in certain Regions where encryption is not available, including China (Beijing) and China (Ningxia). Unique encryption keys are used for each Region.

Deleting AWS WAF Classic resources

You can delete the resources that you create in AWS WAF Classic. See the guidance for each resource type in following sections.

- Deleting a Web ACL (p. 202)
- Adding and deleting rules from an AWS WAF Classic rule group (p. 207)
- Deleting a rule (p. 193)

Identity and access management in AWS WAF Classic

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

Access to AWS WAF Classic requires credentials. Those credentials must have permissions to access AWS resources, such as an AWS WAF Classic 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 Classic to help secure access to your resources.

- Authentication (p. 222)
- Access control (p. 223)

Authentication

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

- **AWS account root user** – When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the

root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

- **IAM user** – An IAM user is an identity within your AWS account that has specific custom permissions (for example, permissions to create a rule in AWS WAF Classic). 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 AWS tools, you must sign the request yourself. AWS WAF Classic 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 an IAM identity that you can create in your account that has specific permissions. An IAM role is similar to an IAM user in that it is an AWS identity with permissions policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session. IAM roles with temporary credentials are useful in the following situations:

  - **Federated user access** – Instead of creating an IAM user, you can use existing 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*.

  - **AWS service access** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. 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 that are running on an EC2 instance and making AWS CLI or 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 that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances 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 Classic resources. For example, you must have permissions to create an AWS WAF Classic *web ACL* or *rule*.

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

AWS Identity and Access Management

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

AWS WAF Classic 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 Classic 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 Classic resources

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

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). 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 Classic resources and operations

In AWS WAF Classic, the resources are *web ACLs* and *rules*. AWS WAF Classic 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 AWS WAF Console	Name in AWS 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 Classic resources, include the ARN of the resource in the `resource` element of your policy. The ARNs for AWS WAF Classic 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 Classic resource.
- *`ID`*: The ID of the AWS WAF Classic 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 Classic provides a set of operations to work with AWS WAF Classic 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 Classic 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 Classic resource to that user, the user can create an AWS WAF Classic resource. However, your account, to which the user belongs, owns the AWS WAF Classic resource.
- If you create an IAM role in your AWS account with permissions to create an AWS WAF Classic resource, anyone who can assume the role can create an AWS WAF Classic resource. Your account, to which the role belongs, owns the AWS WAF Classic resource.

Managing access to resources

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

> **Note**
> These sections discuss using IAM in the context of AWS WAF Classic. 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 Classic 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 Classic 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 Classic 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 Classic, see Using identity-based policies (IAM policies) for AWS WAF Classic (p. 228). 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.

Authorization based on AWS WAF Classic tags

You can attach tags to AWS WAF Classic resources or pass tags in a request to AWS WAF Classic. To control access based on tags, you provide tag information in the condition element of a policy. For more information about tagging your resources, see Working with Tag Editor.

Specifying policy elements: Actions, effects, resources, and principals

For each AWS WAF Classic resource (see AWS WAF Classic resources and operations (p. 225)), the service defines a set of API operations (see AWS WAF Classic API permissions: Actions, resources, and conditions reference (p. 232)). To grant permissions for these API operations, AWS WAF Classic 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 Classic resources and operations (p. 225).
- **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 Classic `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 Classic API actions and the resources that they apply to, see AWS WAF Classic API permissions: Actions, resources, and conditions reference (p. 232).

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 Classic. However, there are general AWS condition keys that you can use as appropriate. For a complete list of AWS keys, see Available Keys for Conditions in the *IAM User Guide*.

Using identity-based policies (IAM policies) for AWS WAF Classic

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

This section 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 Classic 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 Classic resources. For more information, see Overview of managing access permissions to your AWS WAF Classic resources (p. 224).

For a table that shows all the AWS WAF Classic API actions and the resources that they apply to, see AWS WAF Classic API permissions: Actions, resources, and conditions reference (p. 232).

Topics

Permissions required to use the AWS WAF Classic console

The AWS WAF Classic console provides an integrated environment for you to create and manage AWS WAF Classic resources. The console provides many features and workflows that often require permissions

to create an AWS WAF Classic resource in addition to the API-specific permissions that are documented in the AWS WAF Classic API permissions: Actions, resources, and conditions reference (p. 232).
For more information about these additional console permissions, see Customer managed policy examples (p. 229).

AWS managed (predefined) policies for AWS WAF Classic

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 Classic:

- `AWSWAFReadOnlyAccess` – Grants read-only access to AWS WAF Classic resources.
- `AWSWAFFullAccess` – Grants full access to AWS WAF Classic resources.
- `AWSWAFConsoleReadOnlyAccess` – Grants read-only access to the AWS WAF Classic console, which includes resources for AWS WAF and integrated services, such as Amazon CloudFront, Amazon API Gateway, Application Load Balancer, and Amazon CloudWatch.
- `AWSWAFConsoleFullAccess` – Grants full access to the AWS WAF Classic console, which includes resources for AWS WAF and integrated services, such as Amazon CloudFront, Amazon API Gateway, Application Load Balancer, and Amazon CloudWatch.

> **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 Classic 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 Classic 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 on the console. As you attach policies to the user, you can verify that the user can perform various operations on 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 Classic resource, see Creating IAM Roles in the *IAM User Guide*.

Example topics

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 Classic, CloudFront, and CloudWatch

The following policy grants users read-only access to AWS WAF Classic 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 Classic 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 Classic 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 Classic, CloudFront, and CloudWatch

The following policy lets users perform any AWS WAF Classic 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 Classic 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 Classic 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 Classic 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:UpdateDistribution",
            "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 Classic `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 Classic API permissions: Actions, resources, and conditions reference

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

When you set up Access control (p. 223) 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 Classic 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 condition keys in your AWS WAF Classic policies to express conditions. For a complete list of AWS 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 Classic API and required permissions for actions

CreateByteMatchSet

Action: `waf:CreateByteMatchSet`

Resource:

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

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

CreateIPSet

Action: `waf:CreateIPSet`

Resource:

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

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

CreateRule

Action: `waf:CreateRule`

Resource:

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

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

CreateRateBasedRule

Action: `waf:CreateRateBasedRule`

Resource:

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

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

CreateSizeConstraintSet

Action: `waf:CreateSizeConstraintSet`

Resource:

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

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

CreateSqlInjectionMatchSet

Action: `waf:CreateSqlInjectionMatchSet`,

Resource:

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

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

CreateWebACL

Action: `waf:CreateWebACL`

Resource:

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

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

CreateXssMatchSet

Action: `waf:CreateXssMatchSet`

Resource:

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

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

DeleteByteMatchSet

Action: `waf:DeleteByteMatchSet`,

Resource:

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

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

DeleteIPSet

Action: `waf:DeleteIPSet`

Resource:

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

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

DeleteRule

Action: `waf:DeleteRule`

Resource:

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

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

DeleteRateBasedRule

Action: `waf:DeleteRateBasedRule`

Resource:

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

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

DeleteSizeConstraintSet

Action: `waf:DeleteSizeConstraintSet`

Resource:

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

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

DeleteSqlInjectionMatchSet

Action: `waf:DeleteSqlInjectionMatchSet`

Resource:

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

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

DeleteWebACL

Action: `waf:DeleteWebACL`

Resource:

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

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

DeleteXssMatchSet

Action: `waf:DeleteXssMatchSet`

Resource:

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

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

GetByteMatchSet

Action: `waf:GetByteMatchSet`

Resource:

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

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

GetChangeToken

Action: `waf:GetChangeToken`

Resource:

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

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

GetChangeTokenStatus

Action: `waf:GetChangeTokenStatus`

Resource:

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

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

GetIPSet

Action: `waf:GetIPSet`

Resource:

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

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

GetRule

Action: `waf:GetRule`

Resource:

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

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

GetRateBasedRule

Action: `waf:GetRateBasedRule`

Resource:

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

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

GetRateBasedRuleManagedKeys

Action: `waf:GetRateBasedRuleManagedKeys`

Resource:

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

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

GetSampledRequests

Action: `waf:GetSampledRequests`

Resource: Resource depends on the parameters that are specified in the API call. You must have access to the rule or web ACL that corresponds 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 an Application Load Balancer): `arn:aws:waf-regional:region:account-id:rule/example1` or `arn:aws:waf-regional:region:account-id:webacl/example2`

GetSizeConstraintSet

Action: `waf:GetSizeConstraintSet`

Resource:

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

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

GetSqlInjectionMatchSet

Action: `waf:GetSqlInjectionMatchSet`

Resource:

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

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

GetWebACL

Action: `waf:GetWebACL`

Resource:

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

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

GetXssMatchSet

Action: `waf:GetXssMatchSet`

Resource:

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

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

ListByteMatchSets

Action: `waf:ListByteMatchSets`

Resource:

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

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

ListIPSets

Action: `waf:ListIPSets`

Resource:

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

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

ListRules

Action: `waf:ListRules`

Resource:

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

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

ListRateBasedRules

Action: `waf:ListRateBasedRules`

Resource:

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

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

ListSizeConstraintSets

Action: `waf:ListSizeConstraintSets`

Resource:

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

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

ListSqlInjectionMatchSets

Action: `waf:ListSqlInjectionMatchSets`

Resource:

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

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

ListWebACLs

Action: `waf:ListWebACLs`

Resource:

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

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

ListXssMatchSets

Action: `waf:ListXssMatchSets`

Resource:

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

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

UpdateByteMatchSet

Action: `waf:UpdateByteMatchSet`

Resource:

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

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

UpdateIPSet

Action: `waf:UpdateIPSet`

Resource:

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

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

UpdateRule

Action: `waf:UpdateRule`

Resource:

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

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

UpdateRateBasedRule

Action: `waf:UpdateRateBasedRule`

Resource:

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

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

UpdateSizeConstraintSet

Action: `waf:UpdateSizeConstraintSet`

Resource:

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

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

UpdateSqlInjectionMatchSet

Action: `waf:UpdateSqlInjectionMatchSet`

Resource:

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

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

UpdateWebACL

Action: `waf:UpdateWebACL`

Resource:

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

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

UpdateXssMatchSet

Action: `waf:UpdateXssMatchSet`

Resource:

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

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

Using service-linked roles for AWS WAF Classic

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not

migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

AWS WAF Classic uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to AWS WAF Classic. Service-linked roles are predefined by AWS WAF Classic and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up AWS WAF Classic easier because you don't have to manually add the necessary permissions. AWS WAF Classic defines the permissions of its service-linked roles, and unless defined otherwise, only AWS WAF Classic can assume its roles. The defined permissions include the trust policy and the permissions policy. That permissions policy can't be attached to any other IAM entity.

You can delete a service-linked role only after first deleting the role's related resources. This protects your AWS WAF Classic resources because you can't inadvertently remove permission to access the resources.

For information about other services that support service-linked roles, see AWS Services That Work with IAM and look for the services that have **Yes** in the **Service-Linked Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-linked role permissions for AWS WAF Classic

AWS WAF Classic uses the following service-linked roles:

- `AWSServiceRoleForWAFLogging`
- `AWSServiceRoleForWAFRegionalLogging`

AWS WAF Classic uses these service-linked roles to write logs to Amazon Kinesis Data Firehose. These roles are used only if you enable logging in AWS WAF. For more information, see Logging Web ACL traffic information (p. 213).

The `AWSServiceRoleForWAFLogging` and `AWSServiceRoleForWAFRegionalLogging` service-linked roles trust the following services (respectively) to assume the role:

- `waf.amazonaws.com`

  `waf-regional.amazonaws.com`

The permissions policies of the roles allow AWS WAF Classic to complete the following actions on the specified resources:

- Action: `firehose:PutRecord` and `firehose:PutRecordBatch` on Amazon Kinesis Data Firehose data stream resources with a name that starts with "aws-waf-logs-." For example, `aws-waf-logs-us-east-2-analytics`.

You must configure permissions to allow an IAM entity (such as a user, group, or role) to create, edit, or delete a service-linked role. For more information, see Service-Linked Role Permissions in the *IAM User Guide*.

Creating a service-linked role for AWS WAF Classic

You don't need to manually create a service-linked role. When you enable AWS WAF Classic logging on the AWS Management Console, or you make a `PutLoggingConfiguration` request in the AWS WAF Classic CLI or the AWS WAF Classic API, AWS WAF Classic creates the service-linked role for you.

You must have the `iam:CreateServiceLinkedRole` permission to enable logging.

If you delete this service-linked role, and then need to create it again, you can use the same process to recreate the role in your account. When you enable AWS WAF Classic logging, AWS WAF Classic creates the service-linked role for you again.

Editing a service-linked role for AWS WAF Classic

AWS WAF Classic doesn't allow you to edit the `AWSServiceRoleForWAFLogging` and `AWSServiceRoleForWAFRegionalLogging` service-linked roles. After you create a service-linked role, you can't change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a Service-Linked Role in the *IAM User Guide*.

Deleting a service-linked role for AWS WAF Classic

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don't have an unused entity that is not actively monitored or maintained. However, you must clean up the resources for your service-linked role before you can manually delete it.

> **Note**
> If the AWS WAF Classic service is using the role when you try to delete the resources, then the deletion might fail. If that happens, wait for a few minutes and try the operation again.

To delete AWS WAF Classic resources used by the `AWSServiceRoleForWAFLogging` and `AWSServiceRoleForWAFRegionalLogging`

1. On the AWS WAF Classic console, remove logging from every web ACL. For more information, see Logging Web ACL traffic information (p. 213).
2. Using the API or CLI, submit a `DeleteLoggingConfiguration` request for each web ACL that has logging enabled. For more information, see AWS WAF Classic API Reference.

To manually delete the service-linked role using IAM

Use the IAM console, the IAM CLI, or the IAM API to delete the `AWSServiceRoleForWAFLogging` and `AWSServiceRoleForWAFRegionalLogging` service-linked roles. For more information, see Deleting a Service-Linked Role in the *IAM User Guide*.

Supported Regions for AWS WAF Classic service-linked roles

AWS WAF Classic supports using service-linked roles in the following AWS Regions.

Region Name	Region Identity	Support in AWS WAF Classic
US East (N. Virginia)	`us-east-1`	Yes
US East (Ohio)	`us-east-2`	Yes
US West (N. California)	`us-west-1`	Yes
US West (Oregon)	`us-west-2`	Yes
Asia Pacific (Mumbai)	`ap-south-1`	Yes
Asia Pacific (Osaka)	`ap-northeast-3`	Yes

Region Name	Region Identity	Support in AWS WAF Classic
Asia Pacific (Seoul)	ap-northeast-2	Yes
Asia Pacific (Singapore)	ap-southeast-1	Yes
Asia Pacific (Sydney)	ap-southeast-2	Yes
Asia Pacific (Tokyo)	ap-northeast-1	Yes
Canada (Central)	ca-central-1	Yes
Europe (Frankfurt)	eu-central-1	Yes
Europe (Ireland)	eu-west-1	Yes
Europe (London)	eu-west-2	Yes
Europe (Paris)	eu-west-3	Yes
South America (São Paulo)	sa-east-1	Yes

Logging and monitoring in AWS WAF Classic

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS WAF Classic and your AWS solutions. You should collect monitoring data from all parts of your AWS solution so that you can more easily debug a multi-point failure if one occurs. AWS provides several tools for monitoring your AWS WAF Classic resources and responding to potential events:

Amazon CloudWatch Alarms

Using CloudWatch alarms, you watch a single metric over a time period that you specify. If the metric exceeds a given threshold, CloudWatch sends a notification to an Amazon SNS topic or AWS Auto Scaling policy. For more information, see Monitoring with Amazon CloudWatch (p. 370).

AWS CloudTrail Logs

CloudTrail provides a record of actions taken by a user, role, or an AWS service in AWS WAF Classic. Using the information collected by CloudTrail, you can determine the request that was made to AWS WAF Classic, the IP address from which the request was made, who made the request, when it was made, and additional details. For more information, see Logging API calls with AWS CloudTrail (p. 376).

Compliance validation for AWS WAF Classic

Note
This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
For the latest version of AWS WAF, see AWS WAF (p. 6).

Third-party auditors assess the security and compliance of AWS WAF Classic as part of multiple AWS compliance programs, such as SOC, PCI, FedRAMP, HIPAA, and others.

For a list of AWS services in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS WAF Classic is determined by the sensitivity of your data, your organization's compliance objectives, and applicable laws and regulations. If your use of AWS WAF Classic is subject to compliance with standards like HIPAA, PCI, or FedRAMP, AWS provides resources to help:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- Architecting for HIPAA Security and Compliance Whitepaper – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- AWS Config – This AWS service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.
- AWS Well-Architected Framework – The AWS Well-Architected Framework helps you build secure cloud applications.

Resilience in AWS WAF Classic

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

Infrastructure security in AWS WAF Classic

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

As a managed service, AWS WAF Classic is protected by the AWS global network security procedures that are described in the Amazon Web Services: Overview of Security Processes whitepaper.

You use AWS published API calls to access AWS WAF Classic through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

AWS WAF Classic quotas

> **Note**
> This is **AWS WAF Classic** documentation. You should only use this version if you created AWS WAF resources, like rules and web ACLs, in AWS WAF prior to November 2019, and you have not migrated them over to the latest version yet. To migrate your resources, see Migrating your AWS WAF Classic resources to AWS WAF (p. 11).
> **For the latest version of AWS WAF**, see AWS WAF (p. 6).

AWS WAF Classic is subject to the following quotas (formerly referred to as limits).

AWS WAF Classic has default quotas on the number of entities per account per Region. You can request an increase to these.

Resource	Default quota per account per Region
Web ACLs	50
Rules	100
Rate-based-rules	5
Conditions per account per Region	For all conditions except for regex match and geo match, 100 of each condition type. For example, 100 size constraint conditions and 100 IP match conditions. For regex and geo match conditions, see the following table.
Requests per Second	25,000 per web ACL*

*This quota applies only to AWS WAF Classic on an Application Load Balancer. Requests per Second (RPS) quotas for AWS WAF Classic on CloudFront are the same as the RPS quotas support by CloudFront that is described in the CloudFront Developer Guide.

The following quotas on AWS WAF Classic entities can't be changed.

Resource	Quota per account per Region
Rule groups per web ACL	2: 1 customer-created rule group and 1 AWS Marketplace rule group
Rules per web ACL	10
Conditions per rule	10
IP address ranges (in CIDR notation) per IP match condition	10,000 You can update up to 1,000 addresses at a time. The API call `UpdateIPSet` accepts a maximum of 1,000

Resource	Quota per account per Region
	addresses in a single request.
IP addresses blocked per rate-based rule	10,000
Minimum rate-based rule rate limit per 5 minute period	100
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 Classic to inspect the headers in web requests for a specified value	40
In string match conditions, the number of characters in the value that you want AWS WAF Classic to search for	50
Regex match conditions	10
In regex match conditions, the number of characters in the pattern that you want AWS WAF Classic to search for	70
In regex match conditions, the number of patterns per pattern set	10
In regex match conditions, the number of pattern sets per regex condition	1
Pattern sets	5
Geo match conditions	50
Locations per geo match condition	50

AWS WAF Classic has the following fixed quotas on calls per account per Region. These quotas apply to the total calls to the service through any available means, including the console, CLI, AWS CloudFormation, the REST API, and the SDKs. These quotas can't be changed.

Call type	Quota per account per Region
Maximum number of calls to `AssociateWebACL`	1 request every 2 seconds
Maximum number of calls to `DisassociateWebACL`	1 request every 2 seconds
Maximum number of calls to `GetWebACLForResource`	1 request per second
Maximum number of calls to `ListResourcesForWebACL`	1 request per second

Call type	Quota per account per Region
Maximum number of calls to `CreateWebACLMigrationStack`	1 request per second
Maximum number of calls to `GetChangeToken`	10 requests per second
Maximum number of calls to `GetChangeTokenStatus`	1 request per second
Maximum number of calls to any individual `List` action, if no other quota is defined for it	5 requests per second
Maximum number of calls to any individual `Create`, `Put`, `Get`, or `Update` action, if no other quota is defined for it	1 request per second

AWS Firewall Manager

AWS Firewall Manager simplifies your administration and maintenance tasks across multiple accounts and resources for a variety of protections, including AWS WAF, AWS Shield Advanced, Amazon VPC security groups, AWS Network Firewall, and Amazon Route 53 Resolver DNS Firewall. With Firewall Manager, you set up your protections just once and the service automatically applies them across your accounts and resources, even as you add new resources.

Firewall Manager provides these benefits:

- Helps to protect resources across accounts
- Helps to protect all resources of a particular type, such as all Amazon CloudFront distributions
- Helps to protect all resources with specific tags
- Automatically adds protection to resources that are added to your account
- Allows you to subscribe all member accounts in an AWS Organizations organization to AWS Shield Advanced, and automatically subscribes new in-scope accounts that join the organization
- Allows you to apply security group rules to all member accounts or specific subsets of accounts in an AWS Organizations organization, and automatically applies the rules to new in-scope accounts that join the organization
- Lets you use your own rules, or purchase managed rules from AWS Marketplace

Firewall Manager is particularly useful when you want to protect your entire organization rather than a small number of specific accounts and resources, or if you frequently add new resources that you want to protect. Firewall Manager also provides centralized monitoring of DDoS attacks across your organization.

Topics

AWS Firewall Manager pricing

Charges incurred by AWS Firewall Manager are for the underlying services, such as AWS WAF and AWS Config. For more information, see AWS Firewall Manager Pricing.

AWS Firewall Manager prerequisites

This topic shows you how to get ready to administer AWS Firewall Manager. You use one Firewall Manager administrator account to manage all Firewall Manager security policies for your organization in

AWS Organizations. Except where noted, perform the prerequisite steps using the account that you will use as the Firewall Manager administrator.

Before you use Firewall Manager for the first time, perform the following steps in sequence.

Topics

Step 1: Join and configure AWS Organizations

To use Firewall Manager, your account must be a member of the organization in the AWS Organizations service where you want to use your Firewall Manager policies.

> **Note**
> For information about Organizations, see AWS Organizations User Guide.

To establish the required AWS Organizations membership and configuration

1. Choose an account to use as the Firewall Manager administrator for the organization in Organizations.
2. If your chosen account isn't already a member of the organization, have it join. Follow the guidance at Inviting an AWS account to join your organization.
3. AWS Organizations has two available feature sets: *consolidated billing features* and *all features*. To use Firewall Manager, your organization must be enabled for all features. If your organization is configured only for consolidated billing, follow the guidance at Enabling All Features in Your Organization.

Step 2: Set the AWS Firewall Manager administrator account

This procedure uses the account and organization that you chose and configured in the preceding step.

When you set the Firewall Manager administrator account, Firewall Manager automatically sets it as the AWS Organizations Delegated Administrator for Firewall Manager. This allows Firewall Manager to access information about the organizational units (OUs). You can use OUs to specify the scope of your Firewall Manager policies. For more information about setting policy scope, see the guidance for the individual policy types under Creating an AWS Firewall Manager policy (p. 266). For more information about Organizations and management accounts, see Managing the AWS Accounts in Your Organization.

To set the Firewall Manager administrator account

1. Sign in to the AWS Management Console using an existing AWS Organizations management account. You can sign in using the account's root user (not recommended) or another IAM user or IAM role within the account that has equivalent permissions.
2. Open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.
3. Choose **Get started**.

4. Type the ID of the account that you've chosen to use as the Firewall Manager administrator.

   Note
   This account is given permission to create and manage Firewall Manager policies across all accounts within your organization.

5. Choose **Set administrator**.

For more information about managing the Firewall Manager administrator account, see Managing the AWS Firewall Manager administrator (p. 252).

Step 3: Enable AWS Config

To use Firewall Manager, you must enable AWS Config.

Note
You incur charges for your AWS Config settings, according to AWS Config pricing. For more information, see Getting Started with AWS Config.

To enable AWS Config for Firewall Manager

1. Enable AWS Config for each of your AWS Organizations member accounts, including the Firewall Manager administrator account. For more information, see Getting Started with AWS Config.
2. Enable AWS Config for each AWS Region that contains the resources that you want to protect. You can enable AWS Config manually, or you can use the AWS CloudFormation template "Enable AWS Config" at AWS CloudFormation StackSets Sample Templates.

   If you don't want to enable AWS Config for all resources, then you must enable the following according to the type of Firewall Manager policies that you use:

   - **WAF policy** – Enable Config for the resource types CloudFront Distribution, Application Load Balancer (choose **ElasticLoadBalancingV2** from the list), API Gateway, WAF WebACL, WAF Regional WebACL, and WAFv2 WebACL. To enable AWS Config to protect a CloudFront distribution, you must be in the US East (N. Virginia) Region. Other Regions don't have CloudFront as an option.
   - **Shield policy** – Enable Config for the resource types Shield Protection, ShieldRegional Protection, Application Load Balancer, EC2 EIP, WAF WebACL, WAF Regional WebACL, and WAFv2 WebACL.
   - **Security group policy** – Enable Config for the resource types EC2 SecurityGroup, EC2 Instance, and EC2 NetworkInterface.
   - **Network Firewall policy** – Enable Config for the resource types NetworkFirewall FirewallPolicy, NetworkFirewall RuleGroup, EC2 VPC, EC2 InternetGateway, EC2 RouteTable, and EC2 Subnet.
   - **DNS Firewall policy** – Enable Config for the resource types DNSFirewall RuleGroup and EC2 VPC.

Step 4: For Network Firewall and DNS Firewall policies, enable resource sharing

To manage Firewall Manager Network Firewall and DNS Firewall policies, you must enable sharing with AWS Organizations in AWS Resource Access Manager. This allows Firewall Manager to deploy protections across your accounts when you create these policy types.

To enable sharing with AWS Organizations in AWS Resource Access Manager

- Follow the guidance at Enable Sharing with AWS Organizations in the *AWS Resource Access Manager User Guide*.

If you run into problems with resource sharing, see the guidance at Resource sharing for Network Firewall and DNS Firewall policies (p. 299).

Step 5: To use AWS Firewall Manager in Regions that are disabled by default

To use Firewall Manager in a Region that's disabled by default, you must enable the Region for both the management account of your AWS organization and the Firewall Manager administrator account.

For information about Regions that are disabled by default and how to enable them, see Managing AWS Regions in the *AWS General Reference*.

To enable a disabled Region

- For both the Organizations management account and the Firewall Manager administrator account, follow the guidance at Enabling a Region in the *AWS General Reference*.

After you follow these steps, you can configure Firewall Manager to begin protecting your resources. For more information, see Getting started with AWS Firewall Manager AWS WAF policies (p. 254).

Managing the AWS Firewall Manager administrator

You use your Firewall Manager administrator account to manage your Firewall Manager policies. When you set the Firewall Manager administrator account, Firewall Manager automatically sets it as the AWS Organizations Delegated Administrator for Firewall Manager. This allows Firewall Manager to access information about the organizational units (OUs) that you use to specify the scope of your Firewall Manager policies. For more information about Organizations and management accounts, see Managing the AWS Accounts in Your Organization.

To begin using Firewall Manager, you set up your Firewall Manager administrator account and perform other required steps. To do this, follow the guidance under AWS Firewall Manager prerequisites (p. 249).

This topic provides information and guidance for managing your existing administrator account.

Required settings for the Firewall Manager administrator

The Firewall Manager administrator account must have the following settings:

- It must be a member of the organization in AWS Organizations where you want to apply your Firewall Manager policies.
- It must be designated as the Firewall Manager administrator by the Organizations management account for the organization.

Changing the AWS Firewall Manager administrator account

To use AWS Firewall Manager, you must log in to the console with a Firewall Manager administrator account. You can designate only one account in an organization as a Firewall Manager administrator account.

To set up an administrator account for the first time, see AWS Firewall Manager prerequisites (p. 249).

After you designate an account as an administrator account, if you later want to designate a different account as the administrator account, perform the following procedure.

> **Important**
> To designate a different account, you first must revoke administrator privileges from the current administrator account. When you revoke the privileges, all Firewall Manager policies created by that account are deleted. You then must sign into Firewall Manager with the AWS Organizations management account to designate a new administrator account.

To designate a different account as the AWS Firewall Manager administrator account (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.
2. In the navigation pane, choose **Settings**.
3. Choose **Revoke administrator account**.

   > **Important**
   > When you revoke administrator privileges from the current administrator account, all Firewall Manager policies created by that account are deleted.
4. Sign out of the AWS Management Console.
5. Sign in to the AWS Management Console using your AWS Organizations management account. You can sign in using your root user credentials for the account (not recommended) or you can sign in using an IAM user or IAM role within the account that has equivalent permissions.
6. Open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.
7. Choose **Get started**.
8. Type an account ID to associate with Firewall Manager. This account will be the new Firewall Manager administrator account.

   Firewall Manager sets the appropriate permissions for the member account that you provide.

   > **Note**
   > The account is given permission to create and manage AWS WAF rules and rule groups and AWS WAF Classic rules across all accounts within the organization.
9. Choose **Set administrator**.

Disqualifying changes to the AWS Firewall Manager administrator account

Some changes to the AWS Firewall Manager administrator account can disqualify it from remaining the administrator account.

This section describes the changes that can disqualify the Firewall Manager administrator account, and how AWS and Firewall Manager handle these changes.

Account removed from the organization in AWS Organizations

If the AWS Firewall Manager administrator account is removed from the organization in AWS Organizations, it can no longer administer policies for the organization. Firewall Manager takes one of the following actions:

- **Account with no policies** – If the Firewall Manager administrator account has no Firewall Manager policies, Firewall Manager revokes the administrator account.

- **Account with Firewall Manager policies** – If the Firewall Manager administrator account has Firewall Manager policies, Firewall Manager sends an email to inform you of the situation and to provide options that you can take, with the help of your AWS sales account representative.

Account closed

If you close the account that you're using for the AWS Firewall Manager administrator, AWS and Firewall Manager handle the closure as follows:

- AWS revokes the account's administrator access from Firewall Manager and Firewall Manager deactivates any policies that were managed by the administrator account. The protections that were provided by those policies are stopped across the organization.
- AWS retains the Firewall Manager policy data for the account for 90 days from the effective date of the administrator account closure. During this 90-day period, you can reopen the closed account.
  - If you reopen the closed account during the 90-day period, AWS reassigns the account as the Firewall Manager administrator and recovers the Firewall Manager policy data for the account.
  - Otherwise, at the end of the 90-day period, AWS permanently deletes all Firewall Manager policy data for the account.

Getting started with AWS Firewall Manager policies

You can use AWS Firewall Manager to enable a number of different types of security policies. The steps for getting set up are slightly different for each.

Topics

Getting started with AWS Firewall Manager AWS WAF policies

To use AWS Firewall Manager to enable AWS WAF rules across your organization, perform the following steps in sequence.

Topics

Step 1: Complete the prerequisites

There are several mandatory steps to prepare your account for AWS Firewall Manager. Those steps are described in AWS Firewall Manager prerequisites (p. 249). Complete all of the prerequisites before proceeding to Step 2: Create and apply an AWS Firewall Manager AWS WAF policy (p. 255).

Step 2: Create and apply an AWS Firewall Manager AWS WAF policy

A Firewall Manager AWS WAF policy contains the rule groups that you want to apply to your resources. Firewall Manager creates a Firewall Manager web ACL in each account where you apply the policy. The individual account managers can add rules and rule groups to the resulting web ACL, in addition to the rule groups that you define here. For information about Firewall Manager AWS WAF policies, see AWS WAF policies (p. 284).

To create a Firewall Manager AWS WAF policy (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).

2. In the navigation pane, choose **Security policies**.
3. Choose **Create policy**.
4. For **Policy type**, choose **AWS WAF**.
5. For **Region**, choose an AWS Region. To protect Amazon CloudFront distributions, choose **Global**.

   To protect resources in multiple Regions (other than CloudFront distributions), you must create separate Firewall Manager policies for each Region.

6. Choose **Next**.
7. For **Policy name**, enter a descriptive name. Firewall Manager includes the policy name in the names of the web ACLs that it manages. The web ACL names have `FMManagedWebACLV2-` followed by the policy name that you enter here, `-`, and the web ACL creation timestamp, in UTC milliseconds. For example, `FMManagedWebACLV2-MyWAFPolicyName-1621880374078`.
8. Under **Policy rules**, for **First rule groups**, choose **Add rule groups**. Expand the **AWS managed rule groups**. For **Core rule set**, toggle **Add to web ACL**. For **AWS known bad inputs**, toggle **Add to web ACL**. Choose **Add rules**.

   For **Last rule groups**, choose **Add rule groups**. Expand the **AWS managed rule groups** and for the **Amazon IP reputation list**, toggle **Add to web ACL**. Choose **Add rules**.

   Under **First rule groups**, select **Core rule set** and choose **Move down**. AWS WAF evaluates web requests against the **AWS known bad inputs** rule group before it evaluates against the **Core rule set**.

   Note
   You can also create your own AWS WAF rule groups if you want, using the AWS WAF console. Any rule groups that you create show up under **Your rule groups** in the **Describe policy : Add rule groups page**.

   The first and last AWS WAF rule groups that you manage through Firewall Manager have names that begin with `PREFMManaged-` or `POSTFMManaged-`, respectively, followed by the Firewall Manager policy name, and the rule group creation timestamp, in UTC milliseconds. For example, `PREFMManaged-MyWAFPolicyName-1621880555123`.

9. Leave the default action for the web ACL at **Allow**.
10. Leave the **Policy action** at the default, to not automatically remediate noncompliant resources. You can change the option later.
11. Choose **Next**.

12. For **Policy scope**, you provide the settings for the accounts, resource types, and tagging that identify the resources you want to apply the policy to. For this tutorial, leave the **AWS accounts** and **Resources** settings, and choose one or more resource types.
13. Choose **Next**.
14. For **Policy tags**, you can add any identifying tags that you want for the Firewall Manager AWS WAF policy. For more information about tags, see Working with Tag Editor. For this tutorial, you can leave this alone.
15. Choose **Next**.
16. Review the new policy. You can make changes by choosing **Edit** in the area that you want to change. This returns you to the corresponding step in the creation wizard. When you are satisfied with the policy, choose **Create policy**.

Step 3: Clean Up

To avoid extraneous charges, delete any unnecessary policies and resources.

To delete a policy (console)

1. On the **AWS Firewall Manager policies** page, choose the radio button next to the policy name, and then choose **Delete**.
2. In the **Delete** confirmation box, select **Delete all policy resources**, and then choose **Delete** again.

   AWS WAF removes the policy and any associated resources, like web ACLs, that it created in your account. The changes might take a few minutes to propagate to all accounts.

Getting started with AWS Firewall Manager AWS Shield Advanced policies

You can use AWS Firewall Manager to enable AWS Shield Advanced protections across your organization.

> **Important**
> Firewall Manager doesn't support Amazon Route 53 or AWS Global Accelerator. If you need to protect these resources with Shield Advanced, you can't use a Firewall Manager policy. Instead, follow the instructions in Adding AWS Shield Advanced protection to AWS resources (p. 342).

To use Firewall Manager to enable Shield Advanced protection, perform the following steps in sequence.

Topics

Step 1: Complete the prerequisites

There are several mandatory steps to prepare your account for AWS Firewall Manager. Those steps are described in AWS Firewall Manager prerequisites (p. 249). Complete all the prerequisites before proceeding to Step 2: Create and apply an AWS Firewall Manager Shield Advanced policy (p. 257).

Step 2: Create and apply an AWS Firewall Manager Shield Advanced policy

After completing the prerequisites, you create an AWS Firewall Manager Shield Advanced policy. A Firewall Manager Shield Advanced policy contains the accounts and resources that you want to protect with Shield Advanced.

> **Important**
> Firewall Manager does not support Amazon Route 53 or AWS Global Accelerator. If you need to protect these resources with Shield Advanced, you can't use a Firewall Manager policy. Instead, follow the instructions in Adding AWS Shield Advanced protection to AWS resources (p. 342).

To create a Firewall Manager Shield Advanced policy (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   > **Note**
   > For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).

2. In the navigation pane, choose **Security policies**.
3. Choose **Create policy**.
4. For **Policy type**, choose **Shield Advanced**.

   To create a Shield Advanced policy, your Firewall Manager administrator account must be subscribed to Shield Advanced. If you are not subscribed, you are prompted to do so. For more information, see AWS Shield pricing (p. 333).

   > **Note**
   > You don't need to manually subscribe each member account to Shield Advanced. Firewall Manager does this for you as part of creating the policy.

5. For **Region**, choose an AWS Region. To protect Amazon CloudFront resources, choose **Global**.

   To protect resources in multiple Regions (other than CloudFront resources), you must create separate Firewall Manager policies for each Region.

6. Choose **Next**.
7. For **Name**, enter a descriptive name.
8. **AWS accounts affected by this policy** allows you to narrow the scope of your policy by specifying accounts to include or exclude. For this tutorial, choose **Include all accounts under my organization.**
9. Choose the types of resources that you want to protect.

   Firewall Manager doesn't support Amazon Route 53 or AWS Global Accelerator. If you need to protect these resources with Shield Advanced, you can't use a Firewall Manager policy. Instead, follow the instructions in Adding AWS Shield Advanced protection to AWS resources (p. 342).

10. If you want to protect only resources with specific tags, or alternatively exclude resources with specific tags, select **Use tags to include/exclude resources**, enter the tags, and then choose either **Include** or **Exclude**. You can choose only one option.

    If you enter more than one tag (separated by commas), and if a resource has any of those tags, it is considered a match.

    For more information about tags, see Working with Tag Editor.

11. Choose **Create and apply this policy to existing and new resources**.

This option applies Shield Advanced protection to each applicable account within an organization in AWS Organizations, and associates the protection with the specified resources in the accounts. This option also applies the policy to all new resources that match the preceding criteria (resource type and tags).

When you create a Shield policy with auto remediation enabled, for each in scope resource that's not already associated with an AWS WAF web ACL, Firewall Manager associates an empty AWS WAF Classic web ACL. The empty web ACL is used only for Shield monitoring purposes. If you then associate any other web ACL to the resource, Firewall Manager removes the empty web ACL association.

Note
Shield Advanced protects up to 1,000 resources per account.

Alternatively, if you choose **Create but do not apply this policy to existing or new resources**, Firewall Manager doesn't apply Shield Advanced protection to any resources. You must apply the policy to resources later.

12. Choose **Next**.
13. Review the new policy. To make any changes, choose **Previous**. When you are satisfied with the policy, choose **Create policy**.

Continue to Step 3: (Optional) authorize the Shield Response Team (SRT) (p. 258).

Step 3: (Optional) authorize the Shield Response Team (SRT)

One of the benefits of AWS Shield Advanced is support from the Shield Response Team (SRT). When you experience a potential DDoS attack, you can contact the AWS Support Center. If necessary, the Support Center escalates your issue to the SRT. The SRT helps you analyze the suspicious activity and assists you in mitigating the issue. This mitigation often involves creating or updating AWS WAF Classic rules and web ACLs in your account. The SRT 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 setting up AWS Shield Advanced, you proactively provide the SRT with the needed authorization. Providing authorization ahead of time helps prevent mitigation delays in the event of an actual attack.

You authorize and contact the SRT at the account level. That is, the account owner, not the Firewall Manager administrator, must perform the following steps to authorize the SRT to mitigate potential attacks. The Firewall Manager administrator can authorize the SRT only for accounts that they own. Likewise, only the account owner can contact the SRT for support.

Note
To use the services of the SRT, you must be subscribed to the Business Support plan or the Enterprise Support plan.

To authorize the SRT to mitigate potential attacks on your behalf, follow the instructions in Configuring AWS Shield Advanced setup (p. 340). You can change SRT access and permissions at any time by using the same steps.

Continue to Step 4: Configure Amazon SNS notifications and Amazon CloudWatch alarms (p. 258).

Step 4: Configure Amazon SNS notifications and Amazon CloudWatch alarms

You can monitor your protected resources for potential DDoS activity using Amazon SNS. To receive notification of possible attacks, create an Amazon SNS topic for each Region.

To create an Amazon SNS topic in Firewall Manager (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, under **AWS FMS**, choose **Settings**.
3. Choose **Create new topic**.
4. Enter a topic name.
5. Enter an email address that the Amazon SNS messages will be sent to, and then choose **Add email address**.
6. Choose **Update SNS configuration**.

Configure Amazon CloudWatch alarms

Shield Advanced records metrics in CloudWatch that you can monitor. For more information, see AWS Shield Advanced metrics and alarms (p. 373). CloudWatch incurs additional costs. For CloudWatch pricing, see Amazon CloudWatch Pricing.

To create a CloudWatch alarm, follow the instructions in Using Amazon CloudWatch Alarms. By default, Shield Advanced configures CloudWatch to alert you after just one indicator of a potential DDoS event. If needed, you can use the CloudWatch console to change this setting to alert you only after multiple indicators are detected.

Note
In addition to the alarms, you can also use a CloudWatch dashboard to monitor potential DDoS activity. The dashboard collects and processes raw data from Shield Advanced into readable, near real-time metrics. You can use statistics in Amazon CloudWatch to gain a perspective on how your web application or service is performing. For more information, see What is CloudWatch in the *Amazon CloudWatch User Guide*.
For instructions about creating a CloudWatch dashboard, see Monitoring with Amazon CloudWatch (p. 370). For information about specific Shield Advanced metrics that you can add to your dashboard, see AWS Shield Advanced metrics and alarms (p. 373).

You can continue from this step without configuring Amazon SNS notifications or CloudWatch alarms. However, doing so significantly reduces your visibility of possible DDoS events.

As a final step for getting started with Firewall Manager and Shield Advanced, review the global threat dashboard, as described in Step 5: Monitor the global threat dashboard (p. 259).

Step 5: Monitor the global threat dashboard

The global threat dashboard provides a near real-time summary of the global AWS threat landscape. The threat landscape includes the largest attack, the top attack vectors, and the relative number of significant attacks. To view the history of significant DDoS attacks, you can customize the dashboard for different time durations. For more information, see Monitoring threats across AWS (p. 349).

Getting started with AWS Firewall Manager Amazon VPC security group policies

To use AWS Firewall Manager to enable Amazon VPC security groups across your organization, perform the following steps in sequence.

Topics

Step 1: Complete the prerequisites

There are several mandatory steps to prepare your account for AWS Firewall Manager. Those steps are described in AWS Firewall Manager prerequisites (p. 249). Complete all the prerequisites before proceeding to Step 2: Create a security group to use in your policy (p. 260).

Step 2: Create a security group to use in your policy

In this step, you create a security group that you could apply across your organization using Firewall Manager.

> **Note**
> For this tutorial, you won't apply your security group policy to the resources in your organization. You'll just create the policy and see what would happen if you applied the policy's security group to your resources. You do this by disabling automatic remediation on the policy.

If you already have a general security group defined, skip this step and go to Step 3: Create and apply an AWS Firewall Manager common security group policy (p. 260).

To create a security group to use in a Firewall Manager common security group policy

- Create a security group that you could apply to all accounts and resources in your organization, following the guidance under Security Groups for Your VPC in the Amazon VPC User Guide.

  For information on the security group rules options, see Security Group Rules Reference.

You are now ready to go to Step 3: Create and apply an AWS Firewall Manager common security group policy (p. 260).

Step 3: Create and apply an AWS Firewall Manager common security group policy

After completing the prerequisites, you create an AWS Firewall Manager common security group policy. A common security group policy provides a centrally controlled security group for your entire AWS organization. It also defines the AWS accounts and resources that the security group applies to. In addition to common security group policies, Firewall Manager supports content audit security group policies, to manage the security group rules in use in your organization, and usage audit security group policies, to manage unused and redundant security groups. For more information, see Security group policies (p. 287).

For this tutorial, you create a common security group policy and set its action to not automatically remediate. This allows you to see what effect the policy would have without making changes to your AWS organization.

To create a Firewall Manager common security group policy (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   > **Note**
   > For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).

2. In the navigation pane, choose **Security policies**.
3. If you have not met the prerequisites, the console displays instructions about how to fix any issues. Follow the instructions, and then return to this step, to create a common security group policy.
4. Choose **Create policy**.
5. For **Policy type**, choose **Security group**.
6. For **Security group policy type**, choose **Common security groups**.
7. For **Region**, choose an AWS Region.
8. Choose **Next**.
9. For **Policy name**, enter a descriptive name.
10. **Policy rules** allow you to choose how the security groups in this policy are applied and maintained. For this tutorial, leave the options unchecked.
11. Choose **Add primary security group**, select the security group that you created for this tutorial, and choose **Add security group**.
12. For **Policy action**, choose **Identify resources that don't comply with the policy rules, but don't auto remediate.**
13. Choose **Next**.
14. **AWS accounts affected by this policy** allows you to narrow the scope of your policy by specifying accounts to include or exclude. For this tutorial, choose **Include all accounts under my organization.**
15. For **Resource type**, choose one or more types, according to the resources you have defined for your AWS organization.
16. **Resources** allows you to narrow the scope of your policy by specifying resource tags for inclusion or exclusion. To use tagging, you need to first tag your resources. For more information about tagging your resources, see Working with Tag Editor. For this tutorial, choose **Include all resources that match the selected resource type**.
17. Choose **Next**.
18. Review your policy settings. Check to be sure that **Policy actions** is set to **Identify resources that don't comply with the policy rules, but don't auto remediate.** This allows you to review the changes that your policy would have, without making changes at this time.
19. Choose **Create policy**.

    In the **AWS Firewall Manager policies** pane, your policy should be listed. It will probably indicate **Pending** under the accounts headings and it will indicate that **Automatic remediation** is disabled. The creation of a policy can take several minutes. After the **Pending** status is replaced with account counts, you can choose the policy name to explore the compliance status of the accounts and resources. For information, see Viewing compliance information for an AWS Firewall Manager policy (p. 300)
20. When you are finished exploring, if you don't want to keep the policy you created for this tutorial, choose the policy name, choose **Delete**, choose **Clean up resources created by this policy.**, and finally choose **Delete**.

For more information about Firewall Manager security group policies, see Security group policies (p. 287).

Getting started with AWS Firewall Manager Network Firewall policies

To use AWS Firewall Manager to enable an AWS Network Firewall firewall across your organization, perform the following steps in sequence. For information about Firewall Manager Network Firewall policies, see AWS Network Firewall policies (p. 293).

Topics

Step 1: Complete the general prerequisites

There are several mandatory steps to prepare your account for AWS Firewall Manager. Those steps are described in AWS Firewall Manager prerequisites (p. 249). Complete all the prerequisites before proceeding to the next step.

Step 2: Create a Network Firewall rule group to use in your policy

To follow this tutorial, you should be familiar with AWS Network Firewall and know how to configure its rule groups and firewall policies.

You must have at least one rule group in Network Firewall that will be used in your AWS Firewall Manager policy. If you haven't already created a rule group in Network Firewall, do so now. For information about using Network Firewall, see the AWS Network Firewall Developer Guide.

Step 3: Create and apply an AWS Firewall Manager Network Firewall policy

After completing the prerequisites, you create an AWS Firewall Manager Network Firewall policy. A Network Firewall policy provides a centrally controlled AWS Network Firewall firewall for your entire AWS organization. It also defines the AWS accounts and resources that the firewall applies to.

For more information about how Firewall Manager manages your Network Firewall policies, see AWS Network Firewall policies (p. 293).

To create a Firewall Manager Network Firewall policy (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Security policies**.
3. If you haven't met the prerequisites, the console displays instructions about how to fix any issues. Follow the instructions, and then return to this step, to create a Network Firewall policy.
4. Choose **Create security policy**.
5. For **Policy type**, choose **AWS Network Firewall**.
6. For **Region**, choose an AWS Region.
7. Choose **Next**.
8. For **Policy name**, enter a descriptive name.
9. The policy configuration allows you to define the firewall policy. This is the same process as the one you use in the AWS Network Firewall console. You add the rule groups that you want to use in your policy and provide the default stateless actions. For this tutorial, configure this policy as you would a firewall policy in Network Firewall.

Note
Auto remediation happens automatically for AWS Firewall Manager Network Firewall policies, so you won't see an option to choose not to auto remediate here.

10. Choose **Next**.
11. For **Firewall endpoints**, choose **Multiple firewall endpoints**. This option provides high availability for your firewall. When you create the policy, Firewall Manager creates a firewall subnet in each Availability Zone where you have public subnets to protect.
12. For **AWS Network Firewall route configuration**, choose **Monitor** to have Firewall Manager monitor your VPCs for route configuration violations and alert you with remediation suggestions to help you to bring the routes into compliance. Optionally, if you don't want to have your route configurations monitored by Firewall Manager and receive these alerts, choose **Off**.

    Note
    Monitoring provides you with details about non-compliant resources due to faulty route configuration, and suggests remediation actions from the Firewall Manager `GetViolationDetails` API. For example, Network Firewall alerts you if traffic is not routed through the firewall endpoints that are created by your policy.

    Warning
    If you choose **Monitor**, you can't change it to **Off** in the future for the same policy. You must create a new policy.
13. For **Traffic type**, select **Add to firewall policy** to route traffic through the internet gateway.
14. **AWS accounts affected by this policy** allows you to narrow the scope of your policy by specifying accounts to include or exclude. For this tutorial, choose **Include all accounts under my organization**.
15. The **Resource type** for a Network Firewall policy is always **VPC**.
16. **Resources** allows you to narrow the scope of your policy by specifying resource tags for inclusion or exclusion. To use tagging, you need to first tag your resources. For more information about tagging your resources, see Working with Tag Editor. For this tutorial, choose **Include all resources that match the selected resource type**.
17. Choose **Next**.
18. Review your policy settings, and then choose **Create policy**.

    In the **AWS Firewall Manager policies** pane, your policy should be listed. The creation of a policy can take several minutes. Until the creation process is complete, the policy indicates that it's pending. When the policy is ready, the status updates with the count of in-scope accounts. You can choose the policy name to explore the compliance status of the accounts and resources. For information, see Viewing compliance information for an AWS Firewall Manager policy (p. 300)
19. When you are finished exploring, if you don't want to keep the policy that you created for this tutorial, choose the policy name, choose **Delete**, choose **Clean up resources created by this policy.**, and finally choose **Delete**.

For more information about Firewall Manager Network Firewall policies, see AWS Network Firewall policies (p. 293).

Getting started with AWS Firewall Manager DNS Firewall policies

To use AWS Firewall Manager to enable Amazon Route 53 Resolver DNS Firewall across your organization, perform the following steps in sequence. For information about Firewall Manager DNS Firewall policies, see Amazon Route 53 Resolver DNS Firewall policies (p. 298).

Topics

Step 1: Complete the general prerequisites

There are several mandatory steps to prepare your account for AWS Firewall Manager. Those steps are described in AWS Firewall Manager prerequisites (p. 249). Complete all the prerequisites before proceeding to the next step.

Step 2: Create your DNS Firewall rule groups to use in your policy

To follow this tutorial, you should be familiar with Amazon Route 53 Resolver DNS Firewall and know how to configure its rule groups.

You must have least one rule group in DNS Firewall that will be used in your AWS Firewall Manager policy. If you haven't already created a rule group in DNS Firewall, do so now. For information about using DNS Firewall, see Amazon Route 53 Resolver DNS Firewall in the Amazon Route 53 Developer Guide.

Step 3: Create and apply an AWS Firewall Manager DNS Firewall policy

After completing the prerequisites, you create an AWS Firewall Manager DNS Firewall policy. A DNS Firewall policy provides a set of centrally controlled DNS Firewall rule group associations for your entire AWS organization. It also defines the AWS accounts and resources that the firewall applies to.

For more information about how Firewall Manager manages your DNS Firewall rule group associations, see Amazon Route 53 Resolver DNS Firewall policies (p. 298).

To create a Firewall Manager DNS Firewall policy (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.
2. In the navigation pane, choose **Security policies**.
3. If you haven't met the prerequisites, the console displays instructions about how to fix any issues. Follow the instructions, and then return to this step, to create a DNS Firewall policy.
4. Choose **Create security policy**.
5. For **Policy type**, choose **Amazon Route 53 Resolver DNS Firewall**.
6. For **Region**, choose an AWS Region.
7. Choose **Next**.
8. For **Policy name**, enter a descriptive name.
9. The policy configuration allows you to define the DNS Firewall rule group associations that you want to manage from Firewall Manager. You add the rule groups that you want to use in your policy. You can define an association to evaluate first for your VPCs and one to evaluate last. For this tutorial, add one or two rule group associations, depending on your needs.
10. Choose **Next**.
11. **AWS accounts affected by this policy** allows you to narrow the scope of your policy by specifying accounts to include or exclude. For this tutorial, choose **Include all accounts under my organization**.
12. The **Resource type** for a DNS Firewall policy is always **VPC**.

13. **Resources** allows you to narrow the scope of your policy by specifying resource tags for inclusion or exclusion. To use tagging, you need to first tag your resources. For more information about tagging your resources, see Working with Tag Editor. For this tutorial, choose **Include all resources that match the selected resource type**.
14. Choose **Next**.
15. Review your policy settings, and then choose **Create policy**.

    In the **AWS Firewall Manager policies** pane, your policy should be listed. The creation of a policy can take several minutes. Until the creation process is complete, the policy indicates that it's pending. When the policy is ready, the status updates with the count of in-scope accounts. You can choose the policy name to explore the compliance status of the accounts and resources. For information, see Viewing compliance information for an AWS Firewall Manager policy (p. 300)
16. When you are finished exploring, if you don't want to keep the policy that you created for this tutorial, choose the policy name, choose **Delete**, choose **Clean up resources created by this policy.**, and finally choose **Delete**.

For more information about Firewall Manager DNS Firewall policies, see Amazon Route 53 Resolver DNS Firewall policies (p. 298).

Working with AWS Firewall Manager policies

AWS Firewall Manager provides the following types of policies:

- **AWS WAF policy** – Firewall Manager supports AWS WAF and AWS WAF Classic policies. For both versions, you define which resources are protected by the policy.
  - For the AWS WAF policy, you can define a set of rule groups to run first in the web ACL and a set of rule groups to run last. In the accounts where you apply the web ACL, the account owner can add rules and rule groups to run in between the two Firewall Manager rule group sets.
  - For AWS WAF Classic, you create a policy that defines a single rule group.
- **Shield Advanced policy** – This policy applies AWS Shield Advanced protection to specified accounts and resources.
- **Amazon VPC security group policy** – This type of policy gives you control over security groups that are in use throughout your organization in AWS Organizations and lets you enforce a baseline set of rules across your organization.
- **Network Firewall policy** – This policy applies AWS Network Firewall protection to your organization's VPCs.
- **Amazon Route 53 Resolver DNS Firewall policy** – This policy applies DNS Firewall protections to your organization's VPCs.

A Firewall Manager policy is specific to the individual policy type. If you want to enforce multiple policy types across accounts, you can create multiple policies. You can create more than one policy for each type.

If you add a new account to an organization that you created with AWS Organizations, Firewall Manager automatically applies the policy to the resources in that account that are within scope of the policy.

General settings for AWS Firewall Manager policies

AWS Firewall Manager managed policies have some common settings and behaviors. For all, you specify a name and define the scope of the policy, and you can use resource tagging to control policy scope. You can choose to view the accounts and resources that are out of compliance without taking corrective action or to automatically remediate noncompliant resources.

For information about policy scope, see AWS Firewall Manager policy scope (p. 279).

Creating an AWS Firewall Manager policy

The steps for creating a policy vary between the different policy types. Make sure to use the procedure for the type of policy that you need.

> **Important**
> AWS Firewall Manager doesn't support Amazon Route 53 or AWS Global Accelerator. If you want to protect these resources with Shield Advanced, you can't use a Firewall Manager policy. Instead, follow the instructions in Adding AWS Shield Advanced protection to AWS resources (p. 342).

Topics

Creating an AWS Firewall Manager policy for AWS WAF

In a Firewall Manager AWS WAF policy, you can use managed rule groups, which AWS and AWS Marketplace sellers create and maintain for you. You can also create and use your own rule groups. For more information about rule groups, see Rule groups (p. 30).

> **Note**
> Firewall Manager supports the new AWS WAF Bot Control managed rule group. For information about Bot Control in AWS WAF, see AWS WAF Bot Control (p. 94).

If you want to use your own rule groups, create those before you create your Firewall Manager AWS WAF policy. For guidance, see Managing your own rule groups (p. 50). To use an individual custom rule, you must define your own rule group, define your rule within that, and then use the rule group in your policy.

For information about Firewall Manager AWS WAF policies, see AWS WAF policies (p. 284).

To create a Firewall Manager policy for AWS WAF (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   > **Note**
   > For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Security policies**.
3. Choose **Create policy**.
4. For **Policy type**, choose **AWS WAF**.
5. For **Region**, choose an AWS Region. To protect Amazon CloudFront distributions, choose **Global**.

   To protect resources in multiple Regions (other than CloudFront distributions), you must create separate Firewall Manager policies for each Region.
6. Choose **Next**.

7. For **Policy name**, enter a descriptive name. Firewall Manager includes the policy name in the names of the web ACLs that it manages. The web ACL names have `FMManagedWebACLV2-` followed by the policy name that you enter here, –, and the web ACL creation timestamp, in UTC milliseconds. For example, `FMManagedWebACLV2-MyWAFPolicyName-1621880374078`.
8. Under **Policy rules**, add the rule groups that you want AWS WAF to evaluate first and last in the web ACL. The individual account managers can add rules and rule groups in between your first rule groups and your last rule groups. For more information, see AWS WAF policies (p. 284).
9. Set the default action for the web ACL. This is the action that AWS WAF takes when a web request doesn't match any of the rules in the web ACL. For more information, see Deciding on the default action for a web ACL (p. 21).
10. For **Policy action**, if you want to create a web ACL in each applicable account within the organization, but not apply the web ACL to any resources yet, choose **Identify resources that don't comply with the policy rules, but don't auto remediate**. You can change the option later.

    If instead you want to automatically apply the policy to existing in-scope resources, choose **Auto remediate any noncompliant resources**. This option creates a web ACL in each applicable account within the AWS organization and associates the web ACL with the resources in the accounts.

    When you choose **Auto remediate any noncompliant resources**, you can also choose to remove existing web ACL associations from in-scope resources, for the web ACLs that aren't managed by another active Firewall Manager policy. If you choose this option, Firewall Manager first associates the policy's web ACL with the resources, and then removes the prior associations. If a resource has an association with another web ACL that's managed by a different active Firewall Manager policy, this choice doesn't affect that association.
11. Choose **Next**.
12. For **AWS accounts this policy applies to**, choose the option as follows:

    - If you want to apply the policy to all accounts in your organization, leave the default selection, **Include all accounts under my AWS organization**.
    - If you want to apply the policy only to specific accounts or accounts that are in specific AWS Organizations organizational units (OUs), choose **Include only the specified accounts and organizational units**, and then add the accounts and OUs that you want to include. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.
    - If you want to apply the policy to all but a specific set of accounts or AWS Organizations organizational units (OUs), choose **Exclude the specified accounts and organizational units, and include all others**, and then add the accounts and OUs that you want to exclude. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.

    You can only choose one of the options.

    After you apply the policy, Firewall Manager automatically evaluates any new accounts against your settings. For example, if you include only specific accounts, Firewall Manager doesn't apply the policy to any new accounts. As another example, if you include an OU, when you add an account to the OU or to any of its child OUs, Firewall Manager automatically applies the policy to the new account.
13. For **Resource type**, choose the types of resources that you want to protect.
14. For **Resources**, if you want to protect (or exclude) only resources that have specific tags, select the appropriate option, then enter the tags to include or exclude. You can choose only one option. For more information about tags, see Working with Tag Editor.

    If you enter more than one tag, a resource must have all of the tags to be included or excluded.
15. Choose **Next**.

16. For **Policy tags**, add any identifying tags that you want for the Firewall Manager AWS WAF policy. For more information about tags, see Working with Tag Editor.
17. Choose **Next**.
18. Review the new policy. To make any changes, choose **Edit** in the area that you want to change. This returns you to the corresponding step in the creation wizard. When you are satisfied with the policy, choose **Create policy**.

Creating an AWS Firewall Manager policy for AWS WAF Classic

To create a Firewall Manager policy for AWS WAF Classic (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Security policies**.
3. Choose **Create policy**.
4. For **Policy type**, choose **AWS WAF Classic**.
5. If you already created the AWS WAF Classic rule group that you want to add to the policy, choose **Create an AWS Firewall Manager policy and add existing rule groups**. If you want to create a new rule group, choose **Create a Firewall Manager policy and add a new rule group**.
6. For **Region**, choose an AWS Region. To protect Amazon CloudFront resources, choose **Global**.

   To protect resources in multiple Regions (other than CloudFront resources), you must create separate Firewall Manager policies for each Region.
7. Choose **Next**.
8. If you are creating a rule group, follow the instructions in Creating an AWS WAF Classic rule group (p. 206). After you create the rule group, continue with the following steps.
9. Enter a policy name.
10. If you are adding an existing rule group, use the dropdown menu to select a rule group to add, and then choose **Add rule group**.
11. A policy has two possible actions: **Action set by rule group** and **Count**. If you want to test the policy and rule group, set the action to **Count**. This action overrides any *block* action specified by the rules in the rule group. That is, if the policy's action is set to **Count**, those requests are only counted and not blocked. Conversely, if you set the policy's action to **Action set by rule group**, actions of the rule group rules are used. Choose the appropriate action.
12. Choose **Next**.
13. For **AWS accounts this policy applies to**, choose the option as follows:

    - If you want to apply the policy to all accounts in your organization, leave the default selection, **Include all accounts under my AWS organization**.
    - If you want to apply the policy only to specific accounts or accounts that are in specific AWS Organizations organizational units (OUs), choose **Include only the specified accounts and organizational units**, and then add the accounts and OUs that you want to include. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.
    - If you want to apply the policy to all but a specific set of accounts or AWS Organizations organizational units (OUs), choose **Exclude the specified accounts and organizational units, and include all others**, and then add the accounts and OUs that you want to exclude. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.

You can only choose one of the options.

After you apply the policy, Firewall Manager automatically evaluates any new accounts against your settings. For example, if you include only specific accounts, Firewall Manager doesn't apply the policy to any new accounts. As another example, if you include an OU, when you add an account to the OU or to any of its child OUs, Firewall Manager automatically applies the policy to the new account.

14. Choose the type of resource that you want to protect.
15. If you want to protect only resources with specific tags, or alternatively exclude resources with specific tags, select **Use tags to include/exclude resources**, enter the tags, and then choose either **Include** or **Exclude**. You can choose only one option.

    If you enter more than one tag (separated by commas), if a resource has any of those tags, it is considered a match.

    For more information about tags, see Working with Tag Editor.
16. If you want to automatically apply the policy to existing resources, choose **Create and apply this policy to existing and new resources**.

    This option creates a web ACL in each applicable account within an AWS organization and associates the web ACL with the resources in the accounts. This option also applies the policy to all new resources that match the preceding criteria (resource type and tags). Alternatively, if you choose **Create policy but do not apply the policy to existing or new resources**, Firewall Manager creates a web ACL in each applicable account within the organization, but doesn't apply the web ACL to any resources. You must apply the policy to resources later. Choose the appropriate option.
17. For **Replace existing associated web ACLs**, you can choose to remove any web ACL associations that are currently defined for in-scope resources, and then replace them with associations to the web ACLs that you are creating with this policy. By default, Firewall Manager doesn't remove existing web ACL associations before it adds the new ones. If you want to remove the existing ones, choose this option.
18. Choose **Next**.
19. Review the new policy. To make any changes, choose **Edit**. When you are satisfied with the policy, choose **Create and apply policy**.

Creating an AWS Firewall Manager policy for AWS Shield Advanced

To create a Firewall Manager policy for Shield Advanced (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Security policies**.
3. Choose **Create policy**.
4. For **Name**, enter a meaningful name.
5. For **Policy type**, choose **Shield Advanced**.

   To create a Shield Advanced policy, you must be subscribed to Shield Advanced. If you are not subscribed, you are prompted to do so. For more information, see AWS Shield pricing (p. 333).

6. For **Region**, choose an AWS Region. To protect Amazon CloudFront resources, choose **Global**.

   To protect resources in multiple Regions (other than CloudFront resources), you must create separate Firewall Manager policies for each Region.
7. Choose **Next**.
8. For **AWS accounts this policy applies to**, choose the option as follows:

   - If you want to apply the policy to all accounts in your organization, leave the default selection, **Include all accounts under my AWS organization**.
   - If you want to apply the policy only to specific accounts or accounts that are in specific AWS Organizations organizational units (OUs), choose **Include only the specified accounts and organizational units**, and then add the accounts and OUs that you want to include. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.
   - If you want to apply the policy to all but a specific set of accounts or AWS Organizations organizational units (OUs), choose **Exclude the specified accounts and organizational units, and include all others**, and then add the accounts and OUs that you want to exclude. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.

   You can only choose one of the options.

   After you apply the policy, Firewall Manager automatically evaluates any new accounts against your settings. For example, if you include only specific accounts, Firewall Manager doesn't apply the policy to any new accounts. As another example, if you include an OU, when you add an account to the OU or to any of its child OUs, Firewall Manager automatically applies the policy to the new account.
9. Choose the type of resource that you want to protect.

   Firewall Manager does not support Amazon Route 53 or AWS Global Accelerator. If you need to protect these resources with Shield Advanced, you can't use a Firewall Manager policy. Instead, follow the instructions in Adding AWS Shield Advanced protection to AWS resources (p. 342).
10. If you want to protect only resources with specific tags, or alternatively exclude resources with specific tags, select **Use tags to include/exclude resources**, enter the tags, and then choose either **Include** or **Exclude**. You can choose only one option.

    If you enter more than one tag (separated by commas), and if a resource has any of those tags, it is considered a match.

    For more information about tags, see Working with Tag Editor.
11. Choose **Create and apply this policy to existing and new resources**.

    This option applies Shield Advanced protection to each applicable account within an AWS organization, and associates the protection with the specified resources in the accounts. This option also applies the policy to all new resources that match the preceding criteria (resource type and tags). Alternatively, if you choose **Create but do not apply this policy to existing or new resources**, Firewall Manager doesn't apply Shield Advanced protection to any resources. You must apply the policy to resources later.
12. Choose **Next**.
13. Review the new policy. To make any changes, choose **Edit**. When you are satisfied with the policy, choose **Create policy**.

Creating an AWS Firewall Manager common security group policy

For information about how common security group policies work, see Common security group policies (p. 287).

To create a common security group policy, you must have a security group already created in your Firewall Manager administrator account that you want to use as the primary for your policy. You can manage security groups through Amazon Virtual Private Cloud (Amazon VPC) or Amazon Elastic Compute Cloud (Amazon EC2). For information, see Working with Security Groups in the Amazon VPC User Guide.

To create a common security group policy (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Security policies**.
3. Choose **Create policy**.
4. For **Policy type**, choose **Security group**.
5. For **Security group policy type**, choose **Common security groups**.
6. For **Region**, choose an AWS Region.
7. Choose **Next**.
8. For **Policy name**, enter a friendly name.
9. For **Policy rules**, do the following:

   a. From the rules options, choose the restrictions that you want to apply to the security group rules and the resources that are within policy scope.

   b. For **Primary security groups**, choose **Add primary security group**, and then choose the security group that you want to use. Firewall Manager populates the list of primary security groups from all Amazon VPC instances in the Firewall Manager administrator account. The default maximum number of primary security groups for a policy is one. For information about increasing the maximum, see AWS Firewall Manager quotas (p. 322).

   c. For **Policy action**, we recommend creating the policy with the option that doesn't automatically remediate. This allows you to assess the effects of your new policy before you apply it. When you are satisfied that the changes are what you want, then edit the policy and change the policy action to enable automatic remediation of noncompliant resources.
10. Choose **Next**.
11. For **AWS accounts this policy applies to**, choose the option as follows:

    - If you want to apply the policy to all accounts in your organization, leave the default selection, **Include all accounts under my AWS organization**.
    - If you want to apply the policy only to specific accounts or accounts that are in specific AWS Organizations organizational units (OUs), choose **Include only the specified accounts and organizational units**, and then add the accounts and OUs that you want to include. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.
    - If you want to apply the policy to all but a specific set of accounts or AWS Organizations organizational units (OUs), choose **Exclude the specified accounts and organizational units, and include all others**, and then add the accounts and OUs that you want to exclude. Specifying an

OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.

You can only choose one of the options.

After you apply the policy, Firewall Manager automatically evaluates any new accounts against your settings. For example, if you include only specific accounts, Firewall Manager doesn't apply the policy to any new accounts. As another example, if you include an OU, when you add an account to the OU or to any of its child OUs, Firewall Manager automatically applies the policy to the new account.

12. For **Resource type**, choose the types of resources that you want to protect.

    If you choose **EC2 instance**, you can choose to include all elastic network interfaces in each Amazon EC2 instance or just the default interface in each instance. If you have more than one elastic network interface in any in-scope Amazon EC2 instance, choosing the option to include all interfaces allows Firewall Manager to apply the policy to all of them. When you enable automatic remediation, if Firewall Manager can't apply the policy to all elastic network interfaces in an Amazon EC2 instance, it marks the instance as noncompliant.

13. For **Resources**, if you want to apply the policy to all resources within the AWS accounts and resource type parameters, choose **Include all resources that match the selected resource type**. If you want to include or exclude specific resources, use tagging to specify the resources, and then choose the appropriate option and add the tags to the list. You can apply the policy either to all resources except those that have all the tags that you specify, or you can apply it to only those that have all the tags that you specify. For more information about tagging your resources, see Working with Tag Editor.

    Note
    If you enter more than one tag, a resource must have all the tags to be a match.

14. For **Shared VPC resources**, if you want to apply the policy to resources in shared VPCs, in addition to the VPCs that the accounts own, select **Include resources from shared VPCs**.
15. Choose **Next**.
16. Review the policy settings to be sure they're what you want, and then choose **Create policy**.

Firewall Manager creates a replica of the primary security group in every Amazon VPC instance contained within the in-scope accounts up to the supported Amazon VPC maximum quota per account. Firewall Manager associates the replica security groups to the resources that are within policy scope for each in-scope account. For more information about how this policy works, see Common security group policies (p. 287).

Creating an AWS Firewall Manager content audit security group policy

For information about how content audit security group policies work, see Content audit security group policies (p. 289).

For some content audit policy settings, you must provide an audit security group for Firewall Manager to use as a template. For example, you might have an audit security group that contains all of the rules that you don't allow in any security group. You must create these audit security groups using your Firewall Manager administrator account, before you can use them in your policy. You can manage security groups through Amazon Virtual Private Cloud (Amazon VPC) or Amazon Elastic Compute Cloud (Amazon EC2). For information, see Working with Security Groups in the Amazon VPC User Guide.

To create a content audit security group policy (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Security policies**.
3. Choose **Create policy**.
4. For **Policy type**, choose **Security group**.
5. For **Security group policy type**, choose **Auditing and enforcement of security group rules**.
6. For **Region**, choose an AWS Region.
7. Choose **Next**.
8. For **Policy name**, enter a friendly name.
9. For **Policy rules**, choose the managed or custom policy rules option that you want to use.

   a. For **Configure managed audit policy rules**, do the following:

      i. For **Configure security group rules to audit**, select the type of security group rules that you want your audit policy to apply to.

      ii. If you want to do things like restrict the protocols, ports, and CIDR range settings that you allow in your security groups, choose **Audit overly permissive security group rules** and select the options that you want.

      For selections that use protocol lists, you can use existing lists and you can create new lists. For information about protocol lists and how to use them in your policy, see Managed lists (p. 281) and Using managed lists (p. 282).

      iii. If you want to enforce restrictions on what specific applications can do, choose **Audit high risk applications** and select the options that you want.

      The following selections are mutually exclusive: **Applications that can access local CIDR ranges only** and **Applications that can use public CIDR ranges**. You can select at most one of them in any policy.

      For selections that use application lists, you can use existing lists and you can create new lists. For information about application lists and how to use them in your policy, see Managed lists (p. 281) and Using managed lists (p. 282).

      iv. Use the **Overrides** settings to explicitly override other settings in the policy. You can choose to always allow or always deny specific security group rules, regardless of whether they comply with the other options that you've set for the policy.

      For this option, you provide an audit security group as your allowed rules or denied rules template. For **Audit security groups**, choose **Add audit security groups**, and then choose the security group that you want to use. Firewall Manager populates the list of audit security groups from all Amazon VPC instances in the Firewall Manager administrator account. The default maximum quota for the number of audit security groups for a policy is one. For information about increasing the quota, see AWS Firewall Manager quotas (p. 322).

   b. For **Configure custom policy rules**, do the following:

      i. From the rules options, choose whether to allow only the rules defined in the audit security groups or deny all the rules. For information about this choice, see Content audit security group policies (p. 289).

ii. For **Audit security groups**, choose **Add audit security groups**, and then choose the security group that you want to use. Firewall Manager populates the list of audit security groups from all Amazon VPC instances in the Firewall Manager administrator account. The default maximum quota for the number of audit security groups for a policy is one. For information about increasing the quota, see AWS Firewall Manager quotas (p. 322).

iii. For **Policy action**, you must create the policy with the option that doesn't automatically remediate. This allows you to assess the effects of your new policy before you apply it. When you are satisfied that the changes are what you want, edit the policy and change the policy action to enable automatic remediation of noncompliant resources.

10. Choose **Next**.
11. For **AWS accounts this policy applies to**, choose the option as follows:

    - If you want to apply the policy to all accounts in your organization, leave the default selection, **Include all accounts under my AWS organization**.
    - If you want to apply the policy only to specific accounts or accounts that are in specific AWS Organizations organizational units (OUs), choose **Include only the specified accounts and organizational units**, and then add the accounts and OUs that you want to include. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.
    - If you want to apply the policy to all but a specific set of accounts or AWS Organizations organizational units (OUs), choose **Exclude the specified accounts and organizational units, and include all others**, and then add the accounts and OUs that you want to exclude. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.

    You can only choose one of the options.

    After you apply the policy, Firewall Manager automatically evaluates any new accounts against your settings. For example, if you include only specific accounts, Firewall Manager doesn't apply the policy to any new accounts. As another example, if you include an OU, when you add an account to the OU or to any of its child OUs, Firewall Manager automatically applies the policy to the new account.
12. For **Resource type**, choose the types of resource that you want to protect.
13. For **Resources**, if you want to apply the policy to all resources within the AWS accounts and resource type parameters, choose **Include all resources that match the selected resource type**. If you want to include or exclude specific resources, use tagging to specify the resources, and then choose the appropriate option and add the tags to the list. You can apply the policy either to all resources except those that have all the tags that you specify, or you can apply it to only those that have all the tags that you specify. For more information about tagging your resources, see Working with Tag Editor.

    Note
    If you enter more than one tag, a resource must have all the tags to be a match.
14. Choose **Next**.
15. Review the policy settings to be sure they're what you want, and then choose **Create policy**.

Firewall Manager compares the audit security group against the in-scope security groups in your AWS organization, according to your policy rules settings. You can review the policy status in the AWS Firewall Manager policy console. After the policy is created, you can edit it and enable automatic remediation to put your auditing security group policy into effect. For more information about how this policy works, see Content audit security group policies (p. 289).

Creating an AWS Firewall Manager usage audit security group policy

For information about how usage audit security group policies work, see Usage audit security group policies (p. 290).

To create a usage audit security group policy (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Security policies**.
3. Choose **Create policy**.
4. For **Policy type**, choose **Security group**.
5. For **Security group policy type**, choose **Auditing and cleanup of unused and redundant security groups**.
6. For **Region**, choose an AWS Region.
7. Choose **Next**.
8. For **Policy name**, enter a friendly name.
9. For **Policy rules**, choose one or both of the options available.

   - If you choose **Security groups within this policy scope must be used by at least one resource.**, Firewall Manager removes any security groups that it determines are unused. By default, Firewall Manager considers security groups as noncompliant with this policy rule if they are unused for any length of time. You can optionally specify a number of minutes that a security group can exist unused before it is considered noncompliant. If you choose this rule, Firewall Manager runs it last when you save the policy.
   - If you choose **Security groups within this policy scope must be unique.**, Firewall Manager consolidates redundant security groups, so that only one is associated with any resources. If you choose this, Firewall Manager runs it first when you save the policy.
10. For **Policy action**, we recommend creating the policy with the option that doesn't automatically remediate. This allows you to assess the effects of your new policy before you apply it. When you are satisfied that the changes are what you want, then edit the policy and change the policy action to enable automatic remediation of noncompliant resources.
11. Choose **Next**.
12. For **AWS accounts this policy applies to**, choose the option as follows:

    - If you want to apply the policy to all accounts in your organization, leave the default selection, **Include all accounts under my AWS organization**.
    - If you want to apply the policy only to specific accounts or accounts that are in specific AWS Organizations organizational units (OUs), choose **Include only the specified accounts and organizational units**, and then add the accounts and OUs that you want to include. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.
    - If you want to apply the policy to all but a specific set of accounts or AWS Organizations organizational units (OUs), choose **Exclude the specified accounts and organizational units, and include all others**, and then add the accounts and OUs that you want to exclude. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.

You can only choose one of the options.

After you apply the policy, Firewall Manager automatically evaluates any new accounts against your settings. For example, if you include only specific accounts, Firewall Manager doesn't apply the policy to any new accounts. As another example, if you include an OU, when you add an account to the OU or to any of its child OUs, Firewall Manager automatically applies the policy to the new account.

13. For **Resources**, if you want to apply the policy to all resources within the AWS accounts and resource type parameters, choose **Include all resources that match the selected resource type**. If you want to include or exclude specific resources, use tagging to specify the resources, and then choose the appropriate option and add the tags to the list. You can apply the policy either to all resources except those that have all the tags that you specify, or you can apply it to only those that have all the tags that you specify. For more information about tagging your resources, see Working with Tag Editor.

    Note
    If you enter more than one tag, a resource must have all the tags to be a match.
14. Choose **Next**.
15. If you haven't excluded the Firewall Manager administrator account from the policy scope, Firewall Manager prompts you to do this. Doing this leaves the security groups in the Firewall Manager administrator account, which you use for common and audit security group policies, under your manual control. Choose the option you want in this dialogue.
16. Review the policy settings to be sure they're what you want, and then choose **Create policy**.

If you chose to require unique security groups, Firewall Manager scans for redundant security groups in each in-scope Amazon VPC instance. Then, if you chose to require that each security group be used by at least one resource, Firewall Manager scans for security groups that have remained unused for the minutes specified in the rule. You can review the policy status in the AWS Firewall Manager policy console. For more information about how this policy works, see Usage audit security group policies (p. 290).

Creating an AWS Firewall Manager policy for AWS Network Firewall

In a Firewall Manager Network Firewall policy, you use rule groups that you manage in AWS Network Firewall. For information about managing your rule groups, see AWS Network Firewall rule groups in the *Network Firewall Developer Guide*.

For information about Firewall Manager Network Firewall policies, see AWS Network Firewall policies (p. 293).

To create a Firewall Manager policy for AWS Network Firewall (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

    Note
    For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Security policies**.
3. Choose **Create policy**.
4. For **Policy type**, choose **Network Firewall firewall**.
5. For **Region**, choose an AWS Region. To protect resources in multiple Regions, you must create separate policies for each Region.

6. Choose **Next**.
7. For **Policy name**, enter a descriptive name. Firewall Manager includes the policy name in the names of the Network Firewall firewalls and firewall policies that it creates.
8. In the policy configuration, configure the firewall policy as you would in Network Firewall. Add your stateless and stateful rule groups and specify the policy's default actions. For information about Network Firewall firewall policy management, see AWS Network Firewall firewall policies in the *AWS Network Firewall Developer Guide*.

   When you create the Firewall Manager Network Firewall policy, Firewall Manager creates firewall policies for the accounts that are within scope. Individual account managers can add rule groups to the firewall policies, but they can't change the configuration that you provide here.
9. For the **Firewall endpoints** configuration, specify how you want the firewall endpoints to be managed by Firewall Manager. We recommend using multiple endpoints for high availability.
10. If you want to provide the CIDR blocks for Firewall Manager to use for firewall subnets in your VPCs, they must all be /28 CIDR blocks. Enter one block per line. If you omit these, Firewall Manager chooses IP addresses for you from those that are available in the VPCs.

    Note
    Auto remediation happens automatically for AWS Firewall Manager Network Firewall policies, so you won't see an option to choose not to auto remediate here.
11. Choose **Next**.
12. For **AWS accounts this policy applies to**, choose the option as follows:

    - If you want to apply the policy to all accounts in your organization, leave the default selection, **Include all accounts under my AWS organization**.
    - If you want to apply the policy only to specific accounts or accounts that are in specific AWS Organizations organizational units (OUs), choose **Include only the specified accounts and organizational units**, and then add the accounts and OUs that you want to include. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.
    - If you want to apply the policy to all but a specific set of accounts or AWS Organizations organizational units (OUs), choose **Exclude the specified accounts and organizational units, and include all others**, and then add the accounts and OUs that you want to exclude. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.

    You can only choose one of the options.

    After you apply the policy, Firewall Manager automatically evaluates any new accounts against your settings. For example, if you include only specific accounts, Firewall Manager doesn't apply the policy to any new accounts. As another example, if you include an OU, when you add an account to the OU or to any of its child OUs, Firewall Manager automatically applies the policy to the new account.
13. The **Resource type** for Network Firewall policies is **VPC**.
14. For **Resources**, if you want to protect (or exclude) only resources that have specific tags, select the appropriate option, then enter the tags to include or exclude. You can choose only one option. For more information about tags, see Working with Tag Editor.

    If you enter more than one tag, a resource must have all of the tags to be included or excluded.
15. Choose **Next**.
16. For **Policy tags**, add any identifying tags that you want for the Firewall Manager Network Firewall policy. For more information about tags, see Working with Tag Editor.
17. Choose **Next**.

18. Review the new policy. To make any changes, choose **Edit** in the area that you want to change. This returns you to the corresponding step in the creation wizard. When you are satisfied with the policy, choose **Create policy**.

Creating an AWS Firewall Manager policy for Amazon Route 53 Resolver DNS Firewall

In a Firewall Manager DNS Firewall policy, you use rule groups that you manage in Amazon Route 53 Resolver DNS Firewall. For information about managing your rule groups, see Managing rule groups and rules in DNS Firewall in the *Amazon Route 53 Developer Guide*.

For information about Firewall Manager DNS Firewall policies, see Amazon Route 53 Resolver DNS Firewall policies (p. 298).

To create a Firewall Manager policy for Amazon Route 53 Resolver DNS Firewall (console)

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Security policies**.
3. Choose **Create policy**.
4. For **Policy type**, choose **Amazon Route 53 Resolver DNS Firewall**.
5. For **Region**, choose an AWS Region. To protect resources in multiple Regions, you must create separate policies for each Region.
6. Choose **Next**.
7. For **Policy name**, enter a descriptive name.
8. In the policy configuration, add the rule groups that you want DNS Firewall to evaluate first and last among your VPCs' rule group associations. You can add up to two rule groups to the policy.

   When you create the Firewall Manager DNS Firewall policy, Firewall Manager creates the rule group associations, with the association priorities that you've provided, for the VPCs and accounts that are within scope. The individual account managers can add rule group associations in between your first and last associations, but they can't change the associations that you define here. For more information, see Amazon Route 53 Resolver DNS Firewall policies (p. 298).
9. Choose **Next**.
10. For **AWS accounts this policy applies to**, choose the option as follows:

    - If you want to apply the policy to all accounts in your organization, leave the default selection, **Include all accounts under my AWS organization**.
    - If you want to apply the policy only to specific accounts or accounts that are in specific AWS Organizations organizational units (OUs), choose **Include only the specified accounts and organizational units**, and then add the accounts and OUs that you want to include. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.
    - If you want to apply the policy to all but a specific set of accounts or AWS Organizations organizational units (OUs), choose **Exclude the specified accounts and organizational units, and include all others**, and then add the accounts and OUs that you want to exclude. Specifying an OU is the equivalent of specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.

You can only choose one of the options.

After you apply the policy, Firewall Manager automatically evaluates any new accounts against your settings. For example, if you include only specific accounts, Firewall Manager doesn't apply the policy to any new accounts. As another example, if you include an OU, when you add an account to the OU or to any of its child OUs, Firewall Manager automatically applies the policy to the new account.

11. The **Resource type** for DNS Firewall policies is **VPC**.
12. For **Resources**, if you want to protect (or exclude) only resources that have specific tags, select the appropriate option, then enter the tags to include or exclude. You can choose only one option. For more information about tags, see Working with Tag Editor.

    If you enter more than one tag, a resource must have all of the tags to be included or excluded.
13. Choose **Next**.
14. For **Policy tags**, add any identifying tags that you want for the Firewall Manager DNS Firewall policy. For more information about tags, see Working with Tag Editor.
15. Choose **Next**.
16. Review the new policy. To make any changes, choose **Edit** in the area that you want to change. This returns you to the corresponding step in the creation wizard. When you are satisfied with the policy, choose **Create policy**.

Deleting an AWS Firewall Manager policy

You can delete a Firewall Manager policy by performing the following steps.

To delete a policy (console)

1. In the navigation pane, choose **Security policies**.
2. Choose the option next to the policy that you want to delete.
3. Choose **Delete**.

Note
When you delete a Firewall Manager common security group policy, to remove the policy's replica security groups, choose the option to clean up the resources created by the policy. Otherwise, after the primary is deleted, the replicas remain and require manual management in each Amazon VPC instance.

Important
When you delete a Firewall Manager Shield Advanced policy, the policy is deleted, but your accounts remain subscribed to Shield Advanced.

AWS Firewall Manager policy scope

The policy scope defines where the policy applies. You can either apply centrally controlled policies to all of your accounts and resources within your organization in AWS Organizations, or to a subset of your accounts and resources. For instructions on how to set policy scope, see Creating an AWS Firewall Manager policy (p. 266).

Policy scope options in AWS Firewall Manager

When you add a new account or resource to your organization, Firewall Manager automatically assesses it against your settings for each policy and applies the policy based on these settings. For example, you

can choose to apply a policy to all accounts except the account numbers in a specified list; you can also choose to apply a policy only to resources that have all of the tags in a list.

AWS accounts in scope

The settings that you provide to define the AWS accounts affected by the policy determine which of the accounts in your AWS organization to apply the policy to. You can choose to apply the policy in one of the following ways:

- To all accounts in your organization
- To only a specific list of included account numbers and AWS Organizations organizational units (OUs)
- To all except a specific list of excluded account numbers and AWS Organizations organizational units (OUs)

For information about AWS Organizations, see AWS Organizations User Guide.

Resources in scope

Similarly to the settings for accounts in scope, the settings that you provide for resources determine which in-scope resource types to apply the policy to. You can choose one of the following:

- All resources
- Resources that have all of the tags that you specify
- All resources except those that have all of the tags that you specify

For more information about tagging your resources, see Working with Tag Editor.

Policy scope management in AWS Firewall Manager

When policies are in place, Firewall Manager manages them continuously and applies them to new AWS accounts and resources as they are added, in accordance with the policy scope.

How Firewall Manager manages AWS accounts and resources

If an account or resource goes out of scope for any reason, AWS Firewall Manager doesn't automatically remove protections or delete Firewall Manager-managed resources unless you select the **Automatically remove protections from resources that leave the policy scope** check box.

> **Note**
> **Automatically remove protections from resources that leave the policy scope** is not available for AWS Shield Advanced or AWS WAF Classic policies.

Selecting this check box directs AWS Firewall Manager to automatically clean up resources that Firewall Manager manages for accounts when those accounts leave the policy scope. For example, Firewall Manager will disassociate a Firewall Manager-managed web ACL from a protected customer resource when the customer resource leaves the policy scope.

To determine which resources should be removed from protection when a customer resource leaves the policy scope, Firewall Manager follows these guidelines:

- *Default behavior*:
  - The associated AWS Config managed rules are deleted. This behavior is independent of the check box.
  - Any protected resource that goes out of scope remains associated and protected. For example, an Application Load Balancer or API from API Gateway that's associated with a web ACL remains associated with the web ACL, and the protection remains in place.
- *With the **Automatically remove protections from resources that leave the policy scope** check box selected*:

- The associated AWS Config managed rules are deleted. This behavior is independent of the check box.
- Any protected resource that goes out of scope is automatically disassociated and removed from protection when it leaves the policy scope. For example, an Elastic Inference accelerator or Amazon EC2 instance is automatically disassociated from the replicated security group when it leaves the policy scope. The replicated security group and its resources are automatically removed from protection.

Managed lists

Managed application and protocol lists streamline your configuration and management of AWS Firewall Manager content audit security group policies. You use managed lists to define the protocols and applications that your policy allows and disallows. For information about content audit security group policies, see Content audit security group policies (p. 289).

You can use the following types of managed lists in a content audit security group policy:

- **Firewall Manager application lists and protocol lists** – Firewall Manager manages these lists.
  - The application lists include `FMS-Default-Public-Access-Apps-Allowed` and `FMS-Default-Public-Access-Apps-Denied`, which describe commonly used applications that should be allowed or denied to the general public.
  - The protocol lists include `FMS-Default-Protocols-Allowed`, a list of commonly used protocols that should be allowed to the general public. You can use any list that Firewall Manager manages, but you can't edit or delete it.
- **Custom application lists and protocol lists** – You manage these lists. You can create lists of either type with the settings that you need. You have full control over your own custom managed lists, and you can create, edit, and delete them as needed.

  Note
  Currently, Firewall Manager doesn't check references to a custom managed list when you delete it. This means that you can delete a custom managed application list or protocol list even when it is in use by an active policy. This can cause the policy to stop functioning. Delete an application list or protocol list only after you have verified that it isn't referenced by any active polices.

Managed lists are AWS resources. You can tag a custom managed list. You can't tag a Firewall Manager managed list.

Managed list versioning

Custom managed lists don't have versions. When you edit a custom list, policies that reference the list automatically use the updated list.

Firewall Manager managed lists are versioned. The Firewall Manager service team publishes new versions as needed, in order to apply the best security practices to the lists.

When you use a Firewall Manager managed list in a policy, you choose your versioning strategy as follows:

- **Latest available version** – If you don't specify an explicit version setting for the list, then your policy automatically uses the latest version. This is the only option available through the console.
- **Explicit version** – If you specify a version for the list, then your policy uses that version. Your policy remains locked to the version that you specified until you modify the version setting. To specify the version, you must define the policy outside of the console, for example through the CLI or one of the SDKs.

For more information about choosing the version setting for a list, see Using managed lists in your content audit security group policies (p. 282).

Using managed lists in your content audit security group policies

When you create a content audit security group policy, you can choose to use managed audit policy rules. Some of the settings for this option require a managed application list or protocol list. Examples of these settings include protocols that are allowed in security group rules and applications can access the internet.

The following restrictions apply for each policy setting that uses a managed list:

- You can specify at most one Firewall Manager managed list for any setting. By default, you can specify at most one custom list. The custom list limit is a soft quota, so you can request an increase to it. For more information, see AWS Firewall Manager quotas (p. 322).
- In the console, if you select a Firewall Manager managed list, you can't specify the version. The policy will always use the latest version of the list. To specify the version, you must define the policy outside of the console, for example through the CLI or one of the SDKs. For information about versioning for Firewall Manager managed lists, see Managed list versioning (p. 281).

For information about creating a content audit security group policy through the console, see Creating a content audit security group policy (p. 272).

Creating a custom managed application list

To create a custom managed application list

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Application lists**.
3. In the **Application lists** page, choose **Create application list**.
4. In the **Create application list** page, give your list a name. Don't use the prefix `fms-` as this is reserved for Firewall Manager.
5. Specify an application either by providing the protocol and port number or by selecting an application from the **Type** drop down. Give your application specification a name.
6. Choose **Add another** as needed and fill in the application information until you have completed your list.
7. (Optional) Apply tags to your list.
8. Choose **Save** to save your list and return to the **Application lists** page.

Creating a custom managed protocol list

To create a custom managed protocol list

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

Note
For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).

2. In the navigation pane, choose **Protocol lists**.
3. In the **Protocol lists** page, choose **Create protocol list**.
4. In the protocol list creation page, give your list a name. Don't use the prefix `fms-` as this is reserved for Firewall Manager.
5. Specify a protocol.
6. Choose **Add another** as needed and fill in the protocol information until you have completed your list.
7. (Optional) Apply tags to your list.
8. Choose **Save** to save your list and return to the **Protocol lists** page.

Viewing a managed list

To view an application list or protocol list

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Application lists** or **Protocol lists**.

   The page displays all of the lists of the selected type that are available for your use. The lists that Firewall Manager manages have a **Y** in the **ManagedList** column.
3. To see the details of a list, choose its name. The detail page displays the list's content and any tags.

   For Firewall Manager managed lists, you can also see the available versions by selecting the **Version** drop down.

Deleting a custom managed list

You can delete custom managed lists. You can't edit or delete lists that Firewall Manager manages.

Note
Currently, Firewall Manager doesn't check references to a custom managed list when you delete it. This means that you can delete a custom managed application list or protocol list even when it is in use by an active policy. This can cause the policy to stop functioning. Only delete an application list or protocol list after you have verified that it isn't referenced by any active polices.

To delete a custom managed application or protocol list

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. Make sure that the list that you want to delete isn't in use in any of your audit security group policies by doing the following:

a. In the navigation pane, choose **Security policies**.

b. In the **AWS Firewall Manager policies** page, select and edit your audit security groups, and remove any references to the custom list that you want to delete.

   If you delete a custom managed list that's in use in an audit security group policy, the policy that's using it can stop functioning.

3. In the navigation pane, choose **Application lists** or **Protocol lists**, depending on the type of list you want to delete.
4. In the list page, select the custom list that you want to delete and choose **Delete**.

AWS WAF policies

In a Firewall Manager AWS WAF policy, you specify the AWS WAF rule groups that you want to use across your resources. When you apply the policy, in each account that's within policy scope, Firewall Manager creates a web ACL that's managed by Firewall Manager. In the resulting web ACLs, individual account managers can add rules and rule groups, in addition to the rule groups that you defined through Firewall Manager.

When Firewall Manager creates a web ACL for the policy, it names the web ACL `FMManagedWebACLV2-`*`policy name`*`-`*`timestamp`*. The timestamp is in UTC milliseconds. For example, `FMManagedWebACLV2-MyWAFPolicyName-1621880374078`.

AWS Firewall Manager enables sampling and Amazon CloudWatch metrics for the web ACLs and rule groups that it creates for an AWS WAF policy.

Rule groups in AWS Firewall Manager AWS WAF policies

The web ACLs that are managed by Firewall Manager AWS WAF policies contain three sets of rules. These sets provide a higher level of prioritization for the rules and rule groups in the web ACL:

- First rule groups, defined by you in the Firewall Manager AWS WAF policy. AWS WAF evaluates these rule groups first.
- Rules and rule groups that are defined by the account managers in the web ACLs. AWS WAF evaluates any account-managed rules or rule groups next.
- Last rule groups, defined by you in the Firewall Manager AWS WAF policy. AWS WAF evaluates these rule groups last.

Within each of these sets of rules, AWS WAF evaluates rules and rule groups as usual, according to their priority settings within the set.

In the policy's first and last rule groups sets, you can only add rule groups. You can use managed rule groups, which AWS Managed Rules and AWS Marketplace sellers create and maintain for you. You can also manage and use your own rule groups. For more information about all of these options, see Rule groups (p. 30).

Note
Firewall Manager supports the new AWS WAF Bot Control managed rule group. For information about Bot Control in AWS WAF, see AWS WAF Bot Control (p. 94).

If you want to use your own rule groups, you create those before you create your Firewall Manager AWS WAF policy. For guidance, see Managing your own rule groups (p. 50). To use an individual custom rule, you must define your own rule group, define your rule within that, and then use the rule group in your policy.

The first and last AWS WAF rule groups that you manage through Firewall Manager have names that begin with `PREFMManaged-` or `POSTFMManaged-`, respectively, followed by the Firewall Manager policy name, and the rule group creation timestamp, in UTC milliseconds. For example, `PREFMManaged-MyWAFPolicyName-1621880555123`.

For information about how AWS WAF evaluates web requests, see Web ACL rule and rule group evaluation (p. 19).

For the procedure to create a Firewall Manager AWS WAF policy, see Creating an AWS Firewall Manager policy for AWS WAF (p. 266).

Firewall Manager enables sampling and Amazon CloudWatch metrics for the rule groups that you define for the AWS WAF policy.

Individual account owners have complete control over the metrics and sampling configuration for any rule or rule group that they add to the policy's managed web ACLs.

Configuring logging for an AWS Firewall Manager AWS WAF policy

You can enable centralized logging for your AWS WAF policies, to get detailed information about traffic within your organization. Information in the logs includes the time that AWS WAF received the request from your AWS resource, detailed information about the request, and the action for the rule that each request matched from all in-scope accounts. For more information about AWS WAF logging, see Logging web ACL traffic information (p. 107).

> **Note**
> AWS Firewall Manager supports this option for the latest version of AWS WAF, and not for AWS WAF Classic.

When you enable centralized logging on an AWS WAF policy, Firewall Manager creates a web ACL for the policy in the Firewall Manager administrator account as follows:

- Firewall Manager creates the web ACL in the Firewall Manager administrator account regardless of whether the account is in scope of the policy.
- The web ACL has logging enabled, with a log name `FMManagedWebACLV2-Logging`*`policy name`*`-`*`timestamp`*, where the timestamp is the UTC time that the log was enabled for the web ACL, in milliseconds. For example, `FMManagedWebACLV2-LoggingMyWAFPolicyName-1621880565180`. The web ACL has no rule groups and no associated resources.
- You are charged for the web ACL according to the AWS WAF pricing guidelines. For more information, see AWS WAF Pricing.
- Firewall Manager deletes the web ACL when you delete the policy.

> **Note**
> Firewall Manager doesn't modify any existing logging configurations in your organization's member accounts.

You send logs from your policy's web ACLs to an Amazon Kinesis Data Firehose where you've configured a storage destination. After you enable logging, AWS WAF delivers logs for each configured web ACL, through the HTTPS endpoint of Kinesis Data Firehose to the configured storage destination. Before you use it, test your delivery stream to be sure that it has enough throughput to accommodate your organization's logs. For more information about how to create an Amazon Kinesis Data Firehose and review the stored logs, see What Is Amazon Kinesis Data Firehose?

You must have the following permissions to successfully enable logging:

- `iam:CreateServiceLinkedRole`

- `firehose:ListDeliveryStreams`
- `wafv2:PutLoggingConfiguration`

For information about service-linked roles and the `iam:CreateServiceLinkedRole` permission, see Using service-linked roles for AWS WAF (p. 130).

To enable logging for an AWS WAF policy

1. Create an Amazon Kinesis Data Firehose using your Firewall Manager administrator account. Use a name starting with the prefix `aws-waf-logs-`. For example, `aws-waf-logs-firewall-manager-central`. Create the data firehose with a `PUT` source and in the region that you are operating. If you are capturing logs for Amazon CloudFront, create the firehose in US East (N. Virginia). Before you use it, test your delivery stream to be sure that it has enough throughput to accommodate your organization's logs. For more information, see Creating an Amazon Kinesis Data Firehose delivery stream.
2. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
3. In the navigation pane, choose **Security Policies**.
4. Choose the AWS WAF policy that you want to enable logging for. For more information about AWS WAF logging, see Logging web ACL traffic information (p. 107).
5. On the **Policy details** tab, in the **Policy rules** section, choose **Edit**.
6. For **Logging configuration status**, choose **Enabled**.
7. Choose the Kinesis Data Firehose that you created for your logging. You must choose a firehose that begins with `aws-waf-logs-`.
8. (Optional) If you don't want certain fields and their values included in the logs, redact those fields. Choose the field to redact, and then choose **Add**. Repeat as necessary to redact additional fields. The redacted fields appear as `XXX` in the logs. For example, if you redact the **URI** field, the **URI** field in the logs will be `XXX`.
9. (Optional) If you don't want to send all requests to the logs, add your filtering criteria and behavior. Under **Filter logs**, for each filter that you want to apply, choose **Add filter**, then choose your filtering criteria and specify whether you want to keep or drop requests that match the criteria. When you finish adding filters, if needed, modify the **Default logging behavior**.
10. Choose **Next**.
11. Review your settings, then choose **Save** to save your changes to the policy.

To disable logging for an AWS WAF policy

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Security Policies**.
3. Choose the AWS WAF policy that you want to disable logging for.
4. On the **Policy details** tab, in the **Policy rules** section, choose **Edit**.
5. For **Logging configuration status**, choose **Disabled**.
6. Choose **Next**.
7. Review your settings, then choose **Save** to save your changes to the policy.

AWS Shield Advanced policies

This topic covers the ways in which changes to an AWS Firewall Manager-Shield Advanced policy and to related accounts and resources affect your protections and associated charges on your account.

When you create a Shield policy with auto remediation enabled, for each in scope resource that's not already associated with an AWS WAF web ACL, Firewall Manager associates an empty AWS WAF Classic web ACL. The empty web ACL is used only for Shield monitoring purposes. If you then associate any other web ACL to the resource, Firewall Manager removes the empty web ACL association.

If you modify an AWS Firewall Manager-Shield Advanced policy in a way that causes an account or resource to go out of scope of the policy, Firewall Manager no longer monitors the account or resource. However, the account continues to be subscribed to Shield Advanced. The resources continue to be protected by Shield Advanced and they continue to incur the Shield Advanced data transfer charges.

If the tags assigned to a resource change in a way that causes the resource to go out of scope of a Firewall Manager Shield Advanced policy, Firewall Manager no longer monitors the resource. However, the resource continues to be protected by Shield Advanced and it continues to incur Shield Advanced data transfer charges.

If an account that was part of a Firewall Manager Shield Advanced policy leaves the organization, it no longer falls under the scope of the policy and no longer is monitored by Firewall Manager. However, the account remains subscribed to Shield Advanced. Because the account is no longer part of the consolidated billing family, the account will incur a prorated Shield Advanced subscription fee.

Security group policies

You can use AWS Firewall Manager security group policies to manage Amazon Virtual Private Cloud security groups for your organization in AWS Organizations. You can apply centrally controlled security group policies to your entire organization or to a select subset of your accounts and resources. You can also monitor and manage the security group policies that are in use in your organization, with auditing and usage security group policies.

Firewall Manager continuously maintains your policies and applies them to accounts and resources as they are added or updated across your organization. For information about AWS Organizations, see AWS Organizations User Guide. For information about Amazon Virtual Private Cloud security groups, see Security Groups for Your VPC in the Amazon VPC User Guide.

You can use Firewall Manager security group policies to do the following across your AWS organization:

- Apply common security groups to specified accounts and resources.
- Audit security group rules, to locate and remediate noncompliant rules.
- Audit usage of security groups, to clean up unused and redundant security groups.

This section covers how Firewall Manager security groups policies work and provides guidance for using them. For procedures to create security group policies, see Creating an AWS Firewall Manager policy (p. 266).

Common security group policies

With a common security group policy, Firewall Manager provides a centrally controlled association of security groups to accounts and resources across your organization. You specify where and how to apply the policy in your organization.

You can apply common security group policies to the following resource types:

- Amazon Elastic Compute Cloud (Amazon EC2) instance

- Elastic Network Interface
- Application Load Balancer
- Classic Load Balancer

For guidance on creating a common security group policy using the console, see Creating a common security group policy (p. 271).

Shared VPCs

In the policy scope settings for a common security group policy, you can choose to include shared VPCs. This choice includes VPCs that are owned by another account and shared with an in-scope account. VPCs that in-scope accounts own are always included. For information about shared VPCs, see Working with shared VPCs in the Amazon VPC User Guide.

The following caveats apply to including shared VPCs. These are in addition to the general caveats for security group policies at Security group policy limitations (p. 292).

- Firewall Manager replicates the primary security group into the VPCs for each in-scope account. For a shared VPC, Firewall Manager replicates the primary security group once for each in-scope account that the VPC is shared with. This can result in multiple replicas in a single shared VPC.
- When you create a new shared VPC, you won't see it represented in the Firewall Manager security group policy details until after you create at least one resource in the VPC that's within the scope of the policy.
- When you disable shared VPCs in a policy that had shared VPCs enabled, in the shared VPCs, Firewall Manager deletes the replica security groups that aren't associated with any resources. Firewall Manager leaves the remaining replica security groups in place, but stops managing them. Removal of these remaining security groups requires manual management in each shared VPC instance.

Primary security groups

For each common security group policy, you provide AWS Firewall Manager with one or more primary security groups:

- Primary security groups must be created by the Firewall Manager administrator account and can reside in any Amazon VPC instance in the account.
- You manage your primary security groups through Amazon Virtual Private Cloud (Amazon VPC) or Amazon Elastic Compute Cloud (Amazon EC2). For information, see Working with Security Groups in the Amazon VPC User Guide.
- You can name one or more security groups as primaries for a Firewall Manager security group policy. By default, the number of security groups allowed in a policy is one, but you can submit a request to increase it. For information, see AWS Firewall Manager quotas (p. 322).

Policy rules settings

You can choose one or both of the following change control behaviors for the security groups and resources of your common security group policy:

- Identify and revert any changes made by local users to replica security groups.
- Disassociate any other security groups from the AWS resources that are within the policy scope.

Policy creation and management

When you create your common security group policy, Firewall Manager replicates the primary security groups to every Amazon VPC instance within the policy scope, and associates the replicated security

groups to accounts and resources that are in scope of the policy. When you modify a primary security group, Firewall Manager propagates the change to the replicas.

When you delete a common security group policy, you can choose whether to clean up the resources created by the policy. For Firewall Manager common security groups, these resources are the replica security groups. Choose the cleanup option unless you want to manually manage each individual replica after the policy is deleted. For most situations, choosing the cleanup option is the simplest approach.

How replicas are managed

The replica security groups in the Amazon VPC instances are managed like other Amazon VPC security groups. For information, see Security Groups for Your VPC in the Amazon VPC User Guide.

Content audit security group policies

Use AWS Firewall Manager content audit security group policies to check and manage the rules that are in use in your organization's security groups. Content audit security group policies apply to all customer-created security groups in use in your AWS organization, according to the scope that you define in the policy.

For guidance on creating a content audit security group policy using the console, see Creating a content audit security group policy (p. 272).

Policy scope resource type

You can apply content audit security group policies to the following resource types:

- Amazon Elastic Compute Cloud (Amazon EC2) instance
- Elastic Network Interface
- Amazon VPC security group

Security groups are considered in scope of the policy if they explicitly are in scope or if they're associated with resources that are in scope.

Policy rule options

You can use either managed policy rules or custom policy rules for each content audit policy, but not both.

- **Managed policy rules** – In a policy with managed rules, you can use application and protocol lists to specify what's allowed and what's denied by the policy. You can use lists that are managed by Firewall Manager. You can also create and use your own application and protocol lists. For information about these types of lists and your management options for custom lists, see Managed lists (p. 281).
- **Custom policy rules** – In a policy with custom policy rules, you specify an existing security group as the audit security group for your policy. You can use the audit security group rules as a template that defines the rules that are allowed or denied by the policy.

Audit security groups

You must create audit security groups using your Firewall Manager administrator account, before you can use them in your policy. You can manage security groups through Amazon Virtual Private Cloud (Amazon VPC) or Amazon Elastic Compute Cloud (Amazon EC2). For information, see Working with Security Groups in the Amazon VPC User Guide.

A security group that you use for a content audit security group policy is used by Firewall Manager only as a comparison reference for the security groups that are in scope of the policy. Firewall Manager doesn't associate it with any resources in your organization.

The way that you define the rules in the audit security group depends on your choices in the policy rules settings:

- **Managed policy rules** – For managed policy rules settings, you use an audit security group to override other settings in the policy, to explicitly allow or deny rules that otherwise might have another compliancy outcome.
  - If you choose to always *allow* the rules that are defined in the audit security group, any rule that matches one that's defined in the audit security group is considered *compliant* with the policy, regardless of the other policy settings.
  - If you choose to always *deny* the rules that are defined in the audit security group, any rule that matches one that's defined in the audit security group is considered *noncompliant* with the policy, regardless of the other policy settings.
- **Custom policy rules** – For custom policy rules settings, the audit security group provides the example of what is acceptable or not acceptable in the in-scope security group rules:
  - If you choose to *allow* the use of the rules, all in-scope security groups must only have rules that are *within* the allowed range of the policy's audit security group rules. *In this case, the policy's security group rules provide the example of what's acceptable to do.*
  - If you choose to *deny* the use of the rules, all in-scope security groups must only have rules that are *not within* the allowed range of the policy's audit security group rules. *In this case, the policy's security group provides the example of what's not acceptable to do.*

Policy creation and management

When you create an audit security group policy, you must have automatic remediation disabled. The recommended practice is to review the effects of policy creation before enabling automatic remediation. After you review the expected effects, you can edit the policy and enable automatic remediation. When automatic remediation is enabled, Firewall Manager updates or removes rules that are noncompliant in in-scope security groups.

Security groups affected by an audit security group policy

All security groups in your organization that are customer-created are eligible to be in scope of an audit security group policy.

Replica security groups are not customer-created and so aren't eligible to be directly in scope of an audit security group policy. However, they can be updated as a result of the policy's automatic remediation activities. A common security group policy's primary security group is customer-created and can be in scope of an audit security group policy. If an audit security group policy makes changes to a primary security group, Firewall Manager automatically propagates those changes to the replicas.

Usage audit security group policies

Use AWS Firewall Manager usage audit security group policies to monitor your organization for unused and redundant security groups and optionally perform cleanup. When you enable automatic remediation for this policy, Firewall Manager does the following:

1. Consolidates redundant security groups, if you've chosen that option.
2. Removes unused security groups, if you've chosen that option.

You can apply usage audit security group policies to the following resource type:

- Amazon VPC security group

For guidance on creating a usage audit security group policy using the console, see Creating a usage audit security group policy (p. 275).

How Firewall Manager remediates redundant security groups

For security groups to be considered redundant, they must have exactly the same rules set and be in the same Amazon VPC instance. To remediate a redundant security group set, Firewall Manager selects one of the security groups in the set to keep, and then associates it to all resources that are associated with the other security groups in the set. Firewall Manager then disassociates the other security groups from the resources they were associated with, which renders them unused.

> **Note**
> If you have also chosen to remove unused security groups, Firewall Manager does that next. This can result in the removal of the security groups that are in the redundant set.

How Firewall Manager remediates unused security groups

For security groups to be considered unused, they must remain unused by any resource for the minimum number of minutes specified in the policy rule. By default, this number is zero. You can give this a higher setting, in order to allow yourself time to associate new security groups with resources. Firewall Manager remediates unused security groups by deleting them from your account, according to your rules settings.

Default account specification

When you create a usage audit security group policy through the console, Firewall Manager automatically chooses **Exclude the specified accounts and include all others**. The service then puts the Firewall Manager administrator account in the list to exclude. This is the recommended approach, and allows you to manually manage the security groups that belong to the Firewall Manager administrator account.

Best practices for security group policies

This section lists recommendations for managing security groups using AWS Firewall Manager.

Exclude the Firewall Manager administrator account

When you set the policy scope, exclude the Firewall Manager administrator account. When you create a usage audit security group policy through the console, this is the default option.

Start with automatic remediation disabled

For content or usage audit security group policies, start with automatic remediation disabled. Review the policy details information to determine the effects that automatic remediation would have. When you are satisfied that the changes are what you want, edit the policy to enable automatic remediation.

Avoid conflicts if you also use outside sources to manage security groups

If you use a tool or service other than Firewall Manager to manage security groups, take care to avoid conflicts between your settings in Firewall Manager and the settings in your outside source. If you use automatic remediation and your settings conflict, you can create a cycle of conflicting remediation that consumes resources on both sides.

For example, say you configure another service to maintain a security group for a set of AWS resources, and you configure a Firewall Manager policy to maintain a different security group for some or all of the same of resources. If you configure either side to disallow any other security group to be associated with the in-scope resources, that side will remove the security group association that's maintained by the other side. If both sides are configured in this way, you can end up with a cycle of conflicting disassociations and associations.

Additionally, say that you create a Firewall Manager audit policy to enforce a security group configuration that conflicts with the security group configuration from the other service. Remediation applied by the Firewall Manager audit policy can update or delete that security group, putting it out of compliance for the other service. If the other service is configured to monitor and automatically

remediate any problems it finds, it will recreate or update the security group, putting it again out of compliance with the Firewall Manager audit policy. If the Firewall Manager audit policy is configured with automatic remediation, it will again update or delete the outside security group, and so on.

To avoid conflicts like these, create configurations that are mutually exclusive, between Firewall Manager and any outside sources.

You can use tagging to exclude outside security groups from automatic remediation by your Firewall Manager policies. To do this, add one or more tags to the security groups or other resources that are managed by the outside source. Then, when you define the Firewall Manager policy scope, in your resources specification, exclude resources that have the tag or tags that you've added.

Similarly, in your outside tool or service, exclude the security groups that Firewall Manager manages from any management or auditing activities. Either don't import the Firewall Manager resources or use Firewall Manager-specific tagging to exclude them from outside management.

Security group policy limitations

This section lists the limitations for using AWS Firewall Manager security group policies:

- Updating security groups for Amazon EC2 elastic network interfaces that were created using the Fargate service type is not supported. You can, however, update security groups for Amazon ECS elastic network interfaces with the Amazon EC2 service type.
- Firewall Manager doesn't support security groups for Amazon EC2 elastic network interfaces that were created by the Amazon Relational Database Service.
- Updating Amazon ECS elastic network interfaces is possible only for Amazon ECS services that use the rolling update (Amazon ECS) deployment controller. For other Amazon ECS deployment controllers such as CODE_DEPLOY or external controllers, Firewall Manager currently can't update the elastic network interfaces.
- With security groups for Amazon EC2 elastic network interfaces, changes to a security group aren't immediately visible to Firewall Manager. Firewall Manager usually detects changes within several hours, but detection can be delayed as much as six hours.
- Firewall Manager doesn't support updating security groups in elastic network interfaces for Network Load Balancers.
- Firewall Manager doesn’t support security group references in common security group policies.

Security group policy use cases

You can use AWS Firewall Manager common security group policies to automate the host firewall configuration for communication between Amazon VPC instances. This section lists standard Amazon VPC architectures and describes how to secure each using Firewall Manager common security group policies. These security group policies can help you apply a unified set of rules to select resources in different accounts and avoid per-account configurations in Amazon Elastic Compute Cloud and Amazon VPC.

With Firewall Manager common security group policies, you can tag just the EC2 elastic network interfaces that you need for communication with instances in another Amazon VPC. The other instances in the same Amazon VPC are then more secure and isolated.

Use case: Monitoring and controlling requests to Application Load Balancers and Classic Load Balancers

You can use a Firewall Manager common security group policy to define which requests your in-scope load balancers should serve. You can configure this through the Firewall Manager console. Only requests that comply with the security group's inbound rules can reach your load balancers, and the load balancers will only distribute requests that meet the outbound rules.

Use case: Internet-accessible, public Amazon VPC

You can use a Firewall Manager common security group policy to secure a public Amazon VPC, for example, to allow only inbound port 443. This is the same as only allowing inbound HTTPS traffic for a public VPC. You can tag public resources within the VPC (for example, as "PublicVPC"), and then set the Firewall Manager policy scope to only resources with that tag. Firewall Manager automatically applies the policy to those resources.

Use case: Public and Private Amazon VPC instances

You can use the same common security group policy for public resources as recommended in the prior use case for internet-accessible, public Amazon VPC instances. You can use a second common security group policy to limit communication between the public resources and the private ones. Tag the resources in the public and private Amazon VPC instances with something like "PublicPrivate" to apply the second policy to them. You can use a third policy to define the allowed communication between the private resources and other corporation or private Amazon VPC instances. For this policy, you can use another identifying tag on the private resources.

Use case: Hub and spoke Amazon VPC instances

You can use a common security group policy to define communications between the hub Amazon VPC instance and spoke Amazon VPC instances. You can use a second policy to define communication from each spoke Amazon VPC instance to the hub Amazon VPC instance.

Use case: Default network interface for Amazon EC2 instances

You can use a common security group policy to allow only standard communications, for example internal SSH and patch/OS update services, and to disallow other insecure communication.

Use case: Identify resources with open permissions

You can use an audit security group policy to identify all resources within your organization that have permission to communicate with public IP addresses or that have IP addresses that belong to third-party vendors.

AWS Network Firewall policies

You can use AWS Firewall Manager Network Firewall *policies* to manage AWS Network Firewall *firewalls* for your Amazon Virtual Private Cloud *VPCs* across your *organization* in AWS Organizations. You can apply centrally controlled firewalls to your entire organization or to a select subset of your accounts and VPCs.

Network Firewall provides network traffic filtering protections for the public subnets in your VPCs. When you apply the Firewall Manager policy, for each account and VPC that's within policy scope, Firewall Manager creates a Network Firewall firewall and deploys firewall endpoints to VPC subnets, to filter network traffic.

> **Note**
> Firewall Manager Network Firewall policies are Firewall Manager policies that you use to manage Network Firewall protections for your VPCs across your organization.
> The Network Firewall protections are specified in resources in the Network Firewall service that are called firewall policies.

For information about using Network Firewall, see the AWS Network Firewall Developer Guide.

The following sections cover requirements for using Firewall Manager Network Firewall policies and describe how the policies work. For the procedure for creating the policy, see Creating an AWS Firewall Manager policy for AWS Network Firewall (p. 276).

You must enable resource sharing

A Network Firewall policy shares Network Firewall rule groups across the accounts in your organization. For this to work, you must have resource sharing enabled for AWS Organizations. For information about how to enable resource sharing, see Resource sharing for Network Firewall and DNS Firewall policies (p. 299).

You must have your Network Firewall rule groups defined

When you specify a new Network Firewall policy, you define the firewall policy the same as you do when you're using AWS Network Firewall directly. You specify the stateless rule groups to add, default stateless actions, and stateful rule groups. Your rule groups must already exist in the Firewall Manager administrator account for you to include them in the policy. For information about creating Network Firewall rule groups, see AWS Network Firewall rule groups.

How Firewall Manager creates firewall endpoints

Depending on how you configure the policy, Firewall Manager creates a single firewall endpoint or multiple firewall endpoints, for each VPC that's within scope.

- For multiple firewall endpoints, Firewall Manager deploys a firewall endpoint in each Availability Zone where you have a subnet with an internet gateway or a Firewall Manager-created firewall endpoint route in the route table. This is the default option for a Network Firewall policy.
- For a single firewall endpoint, Firewall Manager deploys a firewall endpoint in a single Availability Zone in any subnet that has an internet gateway route. With this option, traffic in other zones needs to cross zone boundaries in order to be filtered by the firewall.

> **Note**
> For both of these options, there must be a subnet associated to a route table that has an IPv4/ prefixlist route in it. Firewall Manager does not check for any other resources.

How Firewall Manager manages your firewall subnets

Firewall subnets are the VPC subnets that Firewall Manager creates for the firewall endpoints that filter your network traffic. Each firewall endpoint must be deployed in a dedicated VPC subnet. Firewall Manager creates at least one firewall subnet in each VPC that's within scope of the policy.

Firewall Manager only creates firewall subnets in Availability Zones that have a subnet with an internet gateway route, or a subnet with a route to the firewall endpoints that Firewall Manager created for their policy. For more information, see VPCs and subnets in the Amazon VPC User Guide.

When you first define a Network Firewall policy, you choose one of the following ways for Firewall Manager to manage the firewall subnets in each of the VPCs that are in scope. You cannot change this choice later.

- Deploy a firewall subnet for every Availability Zone that has public subnets. This is the default behavior. This provides high availability of your traffic filtering protections.
- Deploy a single firewall subnet in one Availability Zone. With this choice, Firewall Manager identifies a zone in the VPC that has the most public subnets and creates the firewall subnet there. The single firewall endpoint filters all network traffic for the VPC. This can reduce firewall costs, but it isn't highly available and it requires traffic from other zones to cross zone boundaries in order to be filtered.

You can provide VPC CIDR blocks for Firewall Manager to use for the firewall subnets or you can leave the choice of firewall endpoint addresses up to Firewall Manager to determine.

- If you don't provide CIDR blocks, Firewall Manager queries your VPCs for available IP addresses to use.
- If you provide a list of CIDR blocks, Firewall Manager searches for new subnets only in the CIDR blocks that you provide. You must use /28 CIDR blocks. For each firewall subnet that Firewall Manager creates, it walks your CIDR block list and uses the first one that it finds that is applicable to the Availability Zone and VPC and has available addresses.

If Firewall Manager can't create a required firewall subnet in an Availability Zone, it marks the subnet as noncompliant with the policy. While the zone is in this state, traffic for the zone must cross zone boundaries in order to be filtered by an endpoint in another zone. This is similar to the single firewall subnet scenario.

How Firewall Manager manages your Network Firewall resources

When you define the policy in Firewall Manager, you provide the network traffic filtering behavior of a standard AWS Network Firewall firewall policy. You add stateless and stateful Network Firewall rule groups and specify default actions for packets that don't match any stateless rules. For information on working with firewall policies in AWS Network Firewall, see the AWS Network Firewall firewall policies.

When you save the Network Firewall policy, Firewall Manager creates a firewall and firewall policy in each VPC that's within scope of the policy. Firewall Manager names these Network Firewall resources by concatenating the following values:

- A fixed string, either `FMManagedNetworkFirewall` or `FMManagedNetworkFirewallPolicy`, depending on the resource type.
- Firewall Manager policy name. This is the name you assign when you create the policy.
- Firewall Manager policy ID. This is the AWS resource ID for the Firewall Manager policy.
- Amazon VPC ID. This is the AWS resource ID for the VPC where Firewall Manager creates the firewall and firewall policy.

The following shows an example name for a firewall that's managed by Firewall Manager:

```
FMManagedNetworkFirewallEXAMPLENameEXAMPLEFirewallManagerPolicyIdEXAMPLEVPCId
```

The following shows an example firewall policy name:

```
FMManagedNetworkFirewallPolicyEXAMPLENameEXAMPLEFirewallManagerPolicyIdEXAMPLEVPCId
```

After you create the policy, account owners in the VPCs can't override your firewall policy settings or your rule groups, but they can add rule groups to the firewall policy that Firewall Manager has created.

How Firewall Manager manages and monitors VPC route tables for your policy

When Firewall Manager creates your firewall endpoints, it also creates the VPC route tables for them. However, Firewall Manager doesn't manage your VPC route tables. You must configure your VPC route tables to direct network traffic to the firewall endpoints that are created by Firewall Manager. Using Amazon VPC ingress routing enhancements, change your routing tables to route traffic through the new firewall endpoints. Your changes must insert the firewall endpoints between the subnets that you want to protect and outside locations. The exact routing that you need to do depends on your architecture and its components.

For information about managing route tables for your VPC, see Route tables in the *Amazon Virtual Private Cloud User Guide*. For information about managing your route tables for Network Firewall, see Route table configurations for AWS Network Firewall in the *AWS Network Firewall Developer Guide*.

When you enable monitoring for a policy, Firewall Manager continuously monitors VPC route configurations and alerts you about traffic that bypasses firewall inspection for that VPC. If a subnet has a firewall endpoint route, Firewall Manager looks for the following routes:

- Routes to send traffic to the Network Firewall endpoint.
- Routes to forward the traffic from the Network Firewall endpoint to the internet gateway.
- Inbound routes from the internet gateway to the Network Firewall endpoint.

- Routes back to the subnet.

If a subnet has a Network Firewall route but there's asymmetric routing in Network Firewall and your internet gateway route table, Firewall Manager reports the subnet as noncompliant. Firewall Manager also detects routes to the internet gateway in the firewall route table that Firewall Manager created, as well as the route table for your subnet, and reports them as noncompliant. Additional routes in the Network Firewall subnet route table and your internet gateway route table are also reported as noncompliant. Depending on the violation type, Firewall Manager suggests remediation actions to bring the route configuration into compliance. Firewall Manager doesn't offer suggestions in all cases. For example, if your customer subnet has a firewall endpoint that was created outside of Firewall Manager, Firewall Manager doesn't suggest remediation actions.

Note

- Firewall Manager does not suggest remediation actions for non-IPv4 routes, such as IPv6 and prefix list routes.
- Calls made using the `DisassociateRouteTable` API call can take up to 12 hours to detect.
- Firewall Manager creates a Network Firewall route table for a subnet that contains the firewall endpoints. Firewall Manager assumes that this route table contains only valid internet gateway and VPC default routes. Any extra or invalid routes in this route table are considered to be noncompliant.

When you configure your Firewall Manager policy, if you choose **Monitor** mode, Firewall Manager provides resource violation and remediation details about your resources. You can use these suggested remediation actions to fix route issues in your route tables. If you choose **Off** mode, Firewall Manager doesn't monitor your route table content for you. With this option, you manage your VPC route tables for yourself. For more information about these resource violations, see Viewing compliance information for an AWS Firewall Manager policy (p. 300).

Warning
If you choose **Monitor** under **AWS Network Firewall route configuration** when creating your policy, you can't turn it off for that policy. However, if you choose **Off**, you can enable it later.

Configuring logging for an AWS Firewall Manager AWS Network Firewall policy

You can enable centralized logging for your Network Firewall policies to get detailed information about traffic within your organization. You can select flow logging to capture network traffic flow, or alert logging to report traffic that matches a rule with the rule action set to `DROP` or `ALERT`. For more information about AWS Network Firewall logging, see Logging network traffic from AWS Network Firewall in the *AWS Network Firewall Developer Guide*.

You send logs from your policy's Network Firewall firewalls to an Amazon S3 bucket. After you enable logging, AWS Network Firewall delivers logs for each configured Network Firewall by updating the firewall settings to deliver logs to your selected Amazon S3 buckets with the reserved AWS Firewall Manager prefix, `<policy-name>-<policy-id>`.

Note
This prefix is used by Firewall Manager to determine whether a logging configuration was added by Firewall Manager, or whether it was added by the account owner. If the account owner attempts to use the reserved prefix for their own custom logging, it is overwritten by the logging configuration in the Firewall Manager policy.

For more information about how to create an Amazon S3 bucket and review the stored logs, see What is Amazon S3? in the *Amazon Simple Storage Service User Guide*.

You must have the following permissions to successfully enable logging:

- `s3:GetBucketPolicy`
- `s3:PutBucketPolicy`

Note that only buckets in the Firewall Manager administrator account may be used for AWS Network Firewall central logging.

When you enable centralized logging on a Network Firewall policy, Firewall Manager takes these actions on your account:

- Firewall Manager updates the permissions on selected S3 buckets to allow for log delivery.
- Firewall Manager creates directories in the S3 bucket for each member account in the scope of the policy. The logs for each account can be found at `<bucket-name>/<policy-name>-<policy-id>/AWSLogs/<account-id>`.

To enable logging for a Network Firewall policy

1. Create an Amazon S3 bucket using your Firewall Manager administrator account. For more information, see Creating a bucket in the *Amazon Simple Storage Service User Guide*.
2. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
3. In the navigation pane, choose **Security Policies**.
4. Choose the Network Firewall policy that you want to enable logging for. For more information about AWS Network Firewall logging, see Logging network traffic from AWS Network Firewall in the *AWS Network Firewall Developer Guide*.
5. On the **Policy details** tab, in the **Policy rules** section, choose **Edit**.
6. To enable and aggregate logs, choose one or more options under **Logging configuration**:

   - **Enable and aggregate flow logs**
   - **Enable and aggregate alert logs**
7. Choose the Amazon S3 bucket where you want your logs to be delivered. You must choose a bucket for each log type that you enable. You can use the same bucket for both log types.
8. (Optional) If you want custom member account-created logging to be replaced with the policy's logging configuration, choose **Override existing logging configuration**.
9. Choose **Next**.
10. Review your settings, then choose **Save** to save your changes to the policy.

To disable logging for a Network Firewall policy

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).
2. In the navigation pane, choose **Security Policies**.
3. Choose the Network Firewall policy that you want to disable logging for.
4. On the **Policy details** tab, in the **Policy rules** section, choose **Edit**.
5. Under **Logging configuration status**, deselect **Enable and aggregate flow logs** and **Enable and aggregate alert logs** if they are selected.

6. Choose **Next**.
7. Review your settings, then choose **Save** to save your changes to the policy.

Amazon Route 53 Resolver DNS Firewall policies

You can use AWS Firewall Manager DNS Firewall policies to manage associations between Amazon Route 53 Resolver DNS Firewall rule groups and your Amazon Virtual Private Cloud *VPCs* across your *organization* in AWS Organizations. You can apply centrally controlled rule groups to your entire organization, or to a select subset of your accounts and VPCs.

DNS Firewall provides filtering and regulation of outbound DNS traffic for your VPCs. You create reusable collections of filtering rules in DNS Firewall rule groups and you associate the rule groups to your VPCs. When you apply the Firewall Manager policy, for each account and VPC that's within policy scope, Firewall Manager creates an association between each DNS Firewall rule group in the policy and each VPC that's within scope of the policy, using the association priority settings that you specify in the Firewall Manager policy.

For information about using DNS Firewall, see Amazon Route 53 Resolver DNS Firewall in the Amazon Route 53 Developer Guide.

The following sections cover requirements for using Firewall Manager DNS Firewall policies and describe how the policies work. For the procedure for creating the policy, see Creating an AWS Firewall Manager policy for Amazon Route 53 Resolver DNS Firewall (p. 278).

You must enable resource sharing

A DNS Firewall policy shares DNS Firewall rule groups across the accounts in your organization. For this to work, you must have resource sharing enabled with AWS Organizations. For information about how to enable resource sharing, see Resource sharing for Network Firewall and DNS Firewall policies (p. 299).

You must have your DNS Firewall rule groups defined

When you specify a new DNS Firewall policy, you define the rule groups the same as you do when you're using Amazon Route 53 Resolver DNS Firewall directly. Your rule groups must already exist in the Firewall Manager administrator account for you to include them in the policy. For information about creating DNS Firewall rule groups, see DNS Firewall rule groups and rules.

You define the lowest and highest priority rule group associations

The DNS Firewall rule group associations that you manage through Firewall Manager DNS Firewall policies contain the lowest priority associations and the highest priority associations for your VPCs. In your policy configuration, these appear as first and last rule groups.

DNS Firewall filters DNS traffic for the VPC in the following order:

1. First rule groups, defined by you in the Firewall Manager DNS Firewall policy. Valid values are between 1 and 99.
2. DNS Firewall rule groups that are associated by individual account managers through DNS Firewall.
3. Last rule groups, defined by you in the Firewall Manager DNS Firewall policy. Valid values are between 9901 and 10000.

How Firewall Manager names the rule group associations that it creates

When you save the DNS Firewall policy, if you enabled autoremediation, Firewall Manager creates a DNS Firewall association between the rule groups that you provided in the policy and the VPCs that are in scope of the policy. Firewall Manager names these associations by concatenating the following values:

- The fixed string, `FMManaged_`.
- The Firewall Manager policy ID. This is the AWS resource ID for the Firewall Manager policy.

The following shows an example name for a firewall that's managed by Firewall Manager:

```
FMManaged_EXAMPLEDNSFirewallPolicyId
```

After you create the policy, account owners in the VPCs can't override your firewall policy settings or your rule group associations. They can associate other DNS Firewall rule groups to the VPCs that are in scope of the DNS Firewall policy. Any associations that are created by the individual account owners must have priority settings between your first and last rule group associations.

Resource sharing for Network Firewall and DNS Firewall policies

To manage Firewall Manager Network Firewall and DNS Firewall policies, you must enable resource sharing with AWS Organizations in AWS Resource Access Manager. This allows Firewall Manager to deploy protections across your accounts when you create these policy types.

To enable resource sharing, follow the instructions at Enable Sharing with AWS Organizations in the *AWS Resource Access Manager User Guide*.

Problems with resource sharing

You might encounter problems with resource sharing, either when you use AWS RAM to enable it, or when you're working on Firewall Manager policies that require it.

Examples of these problems include the following:

- When you follow the instructions to enable sharing, in the AWS RAM console, the choice **Enable sharing with AWS Organizations** is grayed out and not available for selection.
- When you work in Firewall Manager on a policy that requires resource sharing, the policy is marked as noncompliant and you see messages indicating that resource sharing or AWS RAM isn't enabled.

If you encounter problems with resource sharing, use the following procedures to try to enable it.

Try again to enable resource sharing

- Try again to enable sharing using one of the following options:
  - (Option) Through the AWS RAM console, follow the instructions at Enable Sharing with AWS Organizations in the *AWS Resource Access Manager User Guide*.
  - (Option) Using the AWS RAM API, call `EnableSharingWithAwsOrganization`. See the documentation at ram/latest/APIReference/API_EnableSharingWithAwsOrganization.html.

Disable resource sharing and then try again to enable it

1. Disable resource sharing using one of the following options:
   - (Option) Through the AWS RAM console, follow the instructions at Disabling resource sharing with AWS Organizations in the *AWS Resource Access Manager User Guide*.
   - (Option) Using the AWS RAM API, call the AWS Organizations action `DisableAWSServiceAccess`. See the documentation at ram/latest/APIReference/API_DisableAWSServiceAccess.html.

This should successfully disable resource sharing.

2. Try again to enable resource sharing by following the instructions in the previous procedure.

Viewing compliance information for an AWS Firewall Manager policy

Third-party auditors assess the security and compliance of AWS services as part of multiple AWS compliance programs, such as SOC, PCI, FedRAMP, and HIPAA.

To learn whether AWS WAF or other AWS services are in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying baseline environments on AWS that are security and compliance focused.
- Architecting for HIPAA Security and Compliance Whitepaper – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.

  Note
  Not all services are compliant with HIPAA.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- Evaluating Resources with Rules in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.
- AWS Audit Manager – This AWS service helps you continuously audit your AWS usage to simplify how you manage risk and compliance with regulations and industry standards.

For all AWS Firewall Manager policies, you can view the compliance status for accounts and resources that are in scope of the policy. An account or resource is in compliance with a Firewall Manager policy if the settings in the policy are reflected in the settings for the account or resource. Each policy type has its own compliance requirements, which you can tune when you define the policy. For some policies, you can also view detailed violation information for in scope resources, to help you to better understand and manage your security risk.

To view the compliance information for a policy

1. Sign in to the AWS Management Console using your Firewall Manager administrator account, and then open the Firewall Manager console at https://console.aws.amazon.com/wafv2/fmsv2.

   Note
   For information about setting up a Firewall Manager administrator account, see AWS Firewall Manager prerequisites (p. 249).

2. In the navigation pane, choose **Security policies**.
3. Choose a policy. In the **Accounts and resources** tab of the policy page, Firewall Manager lists the accounts in your organization, grouped by those that are within scope of the policy and those that are outside of scope.

   The **Accounts within policy scope** pane lists the compliancy status for each account. A **Compliant** status indicates that the policy has successfully been applied to all of in-scope resources for the account. A **Noncompliant** status indicates that the policy hasn't been applied to one or more of the in-scope resources for the account.
4. Choose an account that's noncompliant. In the account page, Firewall Manager lists the ID and type for each noncompliant resource and the reason that the resource is in violation of the policy.

   Note
   For the resource types `AWS::EC2::NetworkInterface` (ENI) and `AWS::EC2::Instance`, Firewall Manager might show a limited number of noncompliant resources. To list additional noncompliant resources, fix the ones that are initially displayed for the account.
5. If the Firewall Manager policy type is a content audit security group policy, you can access detailed violation information for a resource.

   To view violation details, choose the resource.

   Note
   Resources that Firewall Manager found to be noncompliant before the addition of the detailed resource violation page might not have violation details.

   In the resource page, Firewall Manager lists specific details about the violation, according to resource type.

   - **`AWS::EC2::NetworkInterface` (ENI)** – Firewall Manager displays information about the security group that the resource doesn't comply with. Choose the security group to see more detail about it.
   - **`AWS::EC2::Instance`** – Firewall Manager displays the ENI attached to the EC2 instance that's noncompliant. It also displays information about the security group that the resources don't comply with. Choose the security group to see more detail about it.
   - **`AWS::EC2::SecurityGroup`** – Firewall Manager displays the following violation details:
     - **Noncompliant security group rule** – The rule that's in violation, including its protocol, port range, IP CIDR range, and description.
     - **Referenced rule** – The audit security group rule that the noncompliant security group rule violates, with its details.
     - **Violation reasons** – Explanation of the noncompliance finding.
     - **Remediation action** – Suggested action to take. If Firewall Manager can't determine a safe remediation action, this field is blank.
   - **`AWS::EC2::Subnet`** – This is used for Network Firewall policies. Firewall Manager displays the subnet ID, VPC ID, and Availability Zone. If applicable, Firewall Manager includes additional information about the violation, for example the reason the violation occured, or the ID of the route table a subnet should be associated with. The violation description component contains a description of the expected state of the resource, the current, noncompliant state, and if available, a description of what caused the discrepancy.

     For example, the expected state of a subnet might be “Subnet should contain a AWS Network Firewall subnet in its availability zone”, the current state might be “subnet with id subnet-1234 is missing a Network Firewall subnet in availability zone us-east-1e”, and the description might be “Firewall Manager was unable to create a subnet in this AZ because there are no available CIDR blocks.”

     - **Route management violations** – For Network Firewall policies that use Monitor mode, Firewall Manager displays basic subnet information, as well as expected and actual routes in the subnet,

internet gateway, and Network Firewall subnet route table. Firewall Manager alerts you that there's a violation if the actual routes don't match the expected routes in the route table.

- **Remediation actions for route management violations** – For Network Firewall policies that use Monitor mode, Firewall Manager suggests possible remediation actions on route configurations that have violations.

Example – Route management violation and remediation suggestions

A subnet is expected to send traffic through the firewall endpoints, but the current subnet is sending traffic directly to the internet gateway. This is a route management violation. The suggested remediation in this case might be a list of ordered actions. The first being a recommendation to add the required routes to the Network Firewall subnet's route table to direct outgoing traffic to the internet gateway and to direct incoming traffic for destinations inside the VPC to `local`. The second recommendation is to replace the internet gateway route or the invalid Network Firewall route in the subnet's route table to direct outgoing traffic to the firewall endpoints. The third recommendation is to add required routes to the internet gateway's route table to direct incoming traffic to the firewall endpoints.

- **`AWS::EC2::InternetGateway`** – This is used for Network Firewall policies that have Monitor mode enabled.
  - **Route management violations** – The internet gateway is noncompliant if the internet gateway is not associated with a route table, or if there is an invalid route in the internet gateway route table.
  - **Remediation actions for route management violations** – Firewall Manager suggests possible remediation actions to remedy route management violations.

Example 1 – Route management violation and remediation suggestions

An internet gateway is not associated with a route table. The suggested remediation actions might be a list of ordered actions. The first action is to create a route table. The second action is to associate the route table with the internet gateway. The third action is to add the required route to the internet gateway route table.

Example 2 – Route management violation and remediation suggestions

The internet gateway is associated with a valid route table, but the route is configured improperly. The suggested remediation might be a list of ordered actions. The first suggestion is to remove the invalid route. The second is to add the required route to the internet gateway route table.

- **`AWS::NetworkFirewall::FirewallPolicy`** – This is used for Network Firewall policies. Firewall Manager displays information about a Network Firewall firewall policy that's been modified in a way that makes it noncompliant. The information provides the expected firewall policy and the policy that it found in the customer account, so you can compare stateless and stateful rule groups names and priority settings, custom action names, and default stateless actions settings. The violation description component contains a description of the expected state of the resource, the current, noncompliant state, and if available, a description of what caused the discrepancy.
- **`AWS::EC2::VPC`** – This is used for DNS Firewall policies. Firewall Manager displays information about a VPC that's in scope of a Firewall Manager DNS Firewall policy, and that is noncompliant with the policy. The information provided includes the expected rule groups that are expected to be associated with the VPC and the actual rule groups. The violation description component contains a description of the expected state of the resource, the current, noncompliant state, and if available, a description of what caused the discrepancy.

AWS Firewall Manager findings

AWS Firewall Manager creates findings for resources that are out of compliance and for attacks that it detects and sends them to AWS Security Hub. For information about Security Hub findings, see Findings in AWS Security Hub.

When you use Security Hub and Firewall Manager, Firewall Manager automatically sends your findings to Security Hub. For information about getting started with Security Hub, see Setting Up AWS Security Hub in the AWS Security Hub User Guide.

How do I view my Firewall Manager findings?

To view your Firewall Manager findings in Security Hub, follow the guidance at Working with Findings in Security Hub and create a filter using the following settings:

- Attribute set to **Product Name**.
- Operator set to **EQUALS**.
- Value set to `Firewall Manager`. This setting is case sensitive.

Can I disable this?

You can disable the integration of AWS Firewall Manager findings with Security Hub through the Security Hub console. Choose **Integrations** in the navigation bar, then in the Firewall Manager pane, choose **Disable Integration**. For more information, see the AWS Security Hub User Guide.

AWS Firewall Manager Finding Types

AWS WAF policy findings

You can use Firewall Manager AWS WAF policies to apply AWS WAF rule groups to your resources in AWS Organizations. For more information, see Working with AWS Firewall Manager policies (p. 265).

Resource is missing Firewall Manager managed web ACL.

An AWS resource doesn't have the AWS Firewall Manager managed web ACL association in accordance with the Firewall Manager policy. You can enable Firewall Manager remediation on the policy to correct this.

- Severity – 80
- Status settings – PASSED/FAILED
- Updates – If Firewall Manager performs the remediation action, it will update the finding and the severity will lower from `HIGH` to `INFORMATIONAL`. If you perform the remediation, Firewall Manager will not update the finding.

Firewall Manager managed web ACL has misconfigured rule groups.

The rule groups in a web ACL that's managed by Firewall Manager are not configured correctly, according to the Firewall Manager policy. This means that the web ACL is missing the rule groups that the policy requires. You can enable Firewall Manager remediation on the policy to correct this.

- Severity – 80
- Status settings – PASSED/FAILED
- Updates – If Firewall Manager performs the remediation action, it will update the finding and the severity will lower from `HIGH` to `INFORMATIONAL`. If you perform the remediation, Firewall Manager will not update the finding.

AWS Shield Advanced policy findings

You use Firewall Manager Shield policies to protect accounts and resources AWS Shield Advanced. For more information, see Working with AWS Firewall Manager policies (p. 265).

Resource lacks Shield Advanced protection.

An AWS resource that should have Shield Advanced protection, according to the Firewall Manager policy, doesn't have it. You can enable Firewall Manager remediation on the policy, which will enable the protection for the resource.

- Severity – 60
- Status settings – PASSED/FAILED
- Updates – If Firewall Manager performs the remediation action, it will update the finding and the severity will lower from `HIGH` to `INFORMATIONAL`. If you perform the remediation, Firewall Manager will not update the finding.

Shield Advanced detected attack against monitored resource.

Shield Advanced detected an attack on a protected AWS resource. You can enable Firewall Manager remediation on the policy.

- Severity – 70
- Status settings – None
- Updates – Firewall Manager does not update this finding.

Security group common policy findings

For information about security group common policies, see Security group policies (p. 287).

Resource has misconfigured security group.

Firewall Manager has identified a resource that is missing the Firewall Manager managed security group associations that it should have, according to the Firewall Manager policy. You can enable Firewall Manager remediation on the policy, which creates the associations according to the policy settings.

- Severity – 70
- Status settings – PASSED/FAILED
- Updates – Firewall Manager updates this finding.

Firewall Manager replica security group is out of sync with primary security group.

A Firewall Manager replica security group is out of sync with its primary security group, according to their common security group policy. You can enable Firewall Manager remediation on the policy, which syncs the replica security groups with the primary.

- Severity – 80
- Status settings – PASSED/FAILED
- Updates – Firewall Manager updates this finding.

Security group content audit policy findings

For information about security group content audit policies, see Security group policies (p. 287).

Security group is not in compliance with content audit security group.

A Firewall Manager security group content audit policy has identified a noncompliant security group. This is a customer-created security group that's in scope of the content audit policy and that doesn't comply with the settings defined by the policy and its audit security group. You can enable Firewall Manager remediation on the policy, which modifies the noncompliant security group to bring it into compliance.

- Severity – 70
- Status settings – PASSED/FAILED
- Updates – Firewall Manager updates this finding.

Security group usage audit policy findings

For information about security group usage audit policies, see Security group policies (p. 287).

Firewall Manager found redundant security group.

The Firewall Manager security group usage audit has identified a redundant security group. This is a security group with an identical rules set as another security group within the same Amazon Virtual Private Cloud instance. You can enable Firewall Manager automatic remediation on the usage audit policy, which replaces redundant security groups and with a single security group.

- Severity – 30
- Status settings – None
- Updates – Firewall Manager does not update this finding.

Firewall Manager found unused security group.

The Firewall Manager security group usage audit has identified an unused security group. This is a security group that's not referenced by any Firewall Manager common security group policy. You can enable Firewall Manager automatic remediation on the usage audit policy, which removes unused security groups.

- Severity – 30
- Status settings – None
- Updates – Firewall Manager does not update this finding.

Amazon Route 53 Resolver DNS Firewall policy findings

For information about DNS Firewall policies, see Amazon Route 53 Resolver DNS Firewall policies (p. 298).

Resource is missing DNS Firewall protection

A VPC is missing a DNS Firewall rule group association that's defined in the Firewall Manager DNS Firewall policy. The finding lists the rule group that's specified by the policy.

- Severity – 80

Security in AWS Firewall Manager

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. The effectiveness of our security is regularly tested and verified by third-party auditors as part of the AWS compliance programs. To learn about the compliance programs that apply to Firewall Manager, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your organization's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using Firewall Manager. The following topics show you how to configure Firewall Manager to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Firewall Manager resources.

Topics

Data protection in Firewall Manager

The AWS shared responsibility model applies to data protection in AWS Firewall Manager. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. This content includes the security configuration and management tasks for the AWS services that you use. For

more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form fields such as a **Name** field. This includes when you work with Firewall Manager or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Firewall Manager entities—such as policies—are encrypted at rest, except in certain Regions where encryption is not available, including China (Beijing) and China (Ningxia). Unique encryption keys are used for each Region.

Identity and access management in AWS Firewall Manager

Access to AWS Firewall Manager requires credentials. Those credentials must have permissions to access AWS Firewall Manager resources, like policies. The following sections provide details on how you can use AWS Identity and Access Management (IAM) and Firewall Manager 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 first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.
- **IAM user** – An IAM user is an identity within your AWS account that has specific custom permissions (for example, permissions to create a rule in Firewall Manager). 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 AWS tools, you must sign the request yourself. Firewall Manager 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 an IAM identity that you can create in your account that has specific permissions. An IAM role is similar to an IAM user in that it is an AWS identity with permissions policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session. IAM roles with temporary credentials are useful in the following situations:

  - **Federated user access** – Instead of creating an IAM user, you can use existing 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*.

  - **AWS service access** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. 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 that are running on an EC2 instance and making AWS CLI or 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 that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances 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 Firewall Manager resources. For example, you must have permissions to create a Firewall Manager *policy*.

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

- Overview of managing access permissions to your AWS Firewall Manager resources (p. 309)
- Using identity-based policies (IAM policies) for AWS Firewall Manager (p. 313)
- Firewall Manager required permissions for API actions (p. 317)

AWS Identity and Access Management

Firewall Manager 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 Firewall Manager to control which users in your AWS account can create a new policy.

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 Firewall Manager 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). 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 Firewall Manager resources and operations

In AWS Firewall Manager, the resources are *policy*, *applications list*, and *protocols list*. The Amazon Resource Name (ARN) for Firewall Manager resources has the following format:

```
arn:aws:fms:region:account:resource/ID
```

The following table lists the format for each resource.

Name in AWS Firewall Manager Console	Name in AWS Firewall Manager SDK/ CLI	ARN Format	
Policy	`Policy`	`arn:aws:fms:`*`region`*`:`*`account`*`:policy/`*`ID`*	
Applications list	`AppsList`	`arn:aws:fms:`*`region`*`:`*`account`*`:applications-list/`*`ID`*	
Protocols list	`ProtocolsList`	`arn:aws:fms:`*`region`*`:`*`account`*`:protocols-list/`*`ID`*	

To allow or deny access to a subset of Firewall Manager resources, include the ARNs of the resources in the `resource` element of your policy. 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 Firewall Manager resource.
- *`ID`*: The ID of the Firewall Manager 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 policies for the account `111122223333` in Region `us-east-1`:

```
arn:aws:fms:us-east-1:111122223333:policy/*
```

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

AWS Firewall Manager provides a set of operations to work with Firewall Manager 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 a Firewall Manager 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 a Firewall Manager resource to that user, the user can create a Firewall Manager resource. However, your AWS account, to which the user belongs, owns the Firewall Manager resource.
- If you create an IAM role in your AWS account with permissions to create a Firewall Manager resource, anyone who can assume the role can create a Firewall Manager resource. Your AWS account, to which the role belongs, owns the Firewall Manager resource.

Managing access to resources

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

Note
These sections discuss using IAM in the context of AWS Firewall Manager. 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 Firewall Manager 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 a Firewall Manager 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 `fms:GetPolicy` action on all policies in two specific regions.

```
{
"Version": "2012-10-17",
"Statement": [
   {
        "Sid": "VisualEditor0",
        "Effect": "Deny",
        "Action": "fms:GetPolicy",
        "Resource": [
            "arn:aws:fms:us-east-1:*:policy/*",
            "arn:aws:fms:us-west-2:*:policy/*"
        ],
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/stage": "prod"
            }
        }
```

```
    }
]
}
```

For more information about using identity-based policies with Firewall Manager, see Using identity-based policies (IAM policies) for AWS Firewall Manager (p. 313). 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 Firewall Manager doesn't support resource-based policies.

Specifying policy elements: Actions, effects, resources, and principals

For each AWS Firewall Manager resource (see AWS Firewall Manager resources and operations (p. 309)), the service defines a set of API operations (see Firewall Manager required permissions for API actions (p. 317)). To grant permissions for these API operations, Firewall Manager 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 Firewall Manager resources and operations (p. 309).
- **Action** – You use action keywords to identify resource operations that you want to allow or deny. For example, the `fms:CreatePolicy` permission, coupled with the `wafv2:ListRuleGroups` permission, allows the user permissions to perform the AWS Firewall Manager `CreatePolicy` 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 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 Firewall Manager 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 Firewall Manager API actions and the resources that they apply to, see Firewall Manager required permissions for API actions (p. 317).

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 Firewall Manager. However, there are general AWS condition keys that you can use as appropriate. For a complete list of AWS keys, see Available Keys for Conditions in the *IAM User Guide*.

Using identity-based policies (IAM policies) for AWS Firewall Manager

This section 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 Firewall Manager 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 Firewall Manager resources. For more information, see Overview of managing access permissions to your AWS Firewall Manager resources (p. 309).

For a table that shows all the AWS Firewall Manager API actions and the resources that they apply to, see Firewall Manager required permissions for API actions (p. 317).

Topics

Permissions required to use the AWS Firewall Manager console

The AWS Firewall Manager console provides an integrated environment for you to create and manage Firewall Manager resources. The console provides many features and workflows that often require permissions to create a Firewall Manager resource in addition to the API-specific permissions that are documented in the Firewall Manager required permissions for API actions (p. 317). For more information about these additional console permissions, see Customer managed policy examples (p. 314).

AWS managed (predefined) policies for AWS Firewall Manager

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 Firewall Manager and are grouped by use case scenario:

- `AWSFMAdminFullAccess` – Grants full access to Firewall Manager resources for most situations. If you run in to difficulty creating or managing your Firewall Manager policies with this managed policy, see the following section, Granting full access to AWS Firewall Manager resources (p. 314).
- `AWSFMAdminReadOnlyAccess` – Grants read-only access to all Firewall Manager resources.
- `AWSFMMemberReadOnlyAccess` – Grants read-only access to Firewall Manager member 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 Firewall Manager 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 Firewall Manager resources.

Granting full access to AWS Firewall Manager resources

Follow this guidance if you have difficulty creating or managing your Firewall Manager policies with the managed policy, `AWSFMAdminFullAccess`. The managed policy is described in the previous section.

Use the following policy to grant full administrative access to your account:

```
{
            "Version": "2012-10-17",
            "Statement": [
                  {
                          "Sid": "VisualEditor0",
                          "Effect": "Allow",
                          "Action": [
                                  "ec2:DescribeRegions",
                                  "ec2:DescribeSecurityGroups",
                                  "elasticloadbalancing:*",
                                  "firehose:ListDeliveryStreams",
                                  "fms:*",
                                "network-firewall:ListRuleGroups",
                                "network-firewall:DescribeRuleGroup",
                                "organizations:DescribeOrganization",
                                "organizations:DescribeOrganizationalUnit",
                                "organizations:ListRoots",
                                "organizations:ListChildren",
                                "organizations:ListOrganizationalUnitsForParent",
                                "sns:SetTopicAttributes",
                                "sns:GetTopicAttributes",
                                "sns:CreateTopic",
                                "sns:ListTopics",
                                "sns:Subscribe",
                                "route53resolver:ListFirewallRuleGroups",
                                "route53resolver:GetFirewallRuleGroup",
                                "waf:ListRuleGroups",
                                "waf-regional:ListRuleGroups",
                                "wafv2:ListRuleGroups"
                          ],
                           "Resource": "*"
                    }
                ]
             }
```

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 on the console. As you attach policies to the user, you can verify that the user can perform various operations on 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 Firewall Manager resource, see Creating IAM Roles in the *IAM User Guide*.

Example topics

- Example: Give admin user read-only access to Firewall Manager security groups (p. 315)

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: Give admin user read-only access to Firewall Manager security groups

The following policy grants admin users read-only access to Firewall Manager security groups and policies. These users can't create, update, or delete the Firewall Manager resources.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "fms:Get*",
                "fms:List*",
                "ec2:DescribeSecurityGroups"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

AWS managed policies for AWS Firewall Manager

To add permissions to users, groups, and roles, it is easier to use AWS managed policies than to write policies yourself. It takes time and expertise to create IAM customer managed policies that provide your team with only the permissions they need. To get started quickly, you can use our AWS managed policies. These policies cover common use cases and are available in your AWS account. For more information about AWS managed policies, see AWS managed policies in the *IAM User Guide*.

AWS services maintain and update AWS managed policies. You can't change the permissions in AWS managed policies. Services occasionally add additional permissions to an AWS managed policy to support new features. This type of update affects all identities (users, groups, and roles) where the policy is attached. Services are most likely to update an AWS managed policy when a new feature is launched or when new operations become available. Services do not remove permissions from an AWS managed policy, so policy updates won't break your existing permissions.

Additionally, AWS supports managed policies for job functions that span multiple services. For example, the **ReadOnlyAccess** AWS managed policy provides read-only access to all AWS services and resources. When a service launches a new feature, AWS adds read-only permissions for new operations and resources. For a list and descriptions of job function policies, see AWS managed policies for job functions in the *IAM User Guide*.

Firewall Manager updates to AWS managed policies

View details about updates to AWS managed policies for Firewall Manager since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the Firewall Manager document history page at Document history (p. 393).

Change	Description	Date
`FMSServiceRolePolicy` – New permissions for security group and AWS Network Firewall policies	Added new permissions to enable centralized logging for AWS Network Firewall policies. Additionally, read-only Amazon EC2 permissions were added to support changes to the Config service that impact how AWS Firewall Manager queries resources for security group policies.	September 29, 2021
`FMSServiceRolePolicy` – ARN formats for AWS WAF resources	Updated the `FMSServiceRolePolicy` to standardize the ARN formats for AWS WAF resources. The updated ARN formats are `arn:aws:waf:*:*:*` and `arn:aws:waf-regional:*:*:*`.	August 12, 2021
`FMSServiceRolePolicy` – Additional regions in China	AWS Firewall Manager has enabled `FMSServiceRolePolicy` for the BJS and ZHY regions in China.	August 12, 2021
`FMSServiceRolePolicy` – Update to the existing policy	Added new permissions to allow AWS Firewall Manager to manage Amazon Route 53 Resolver DNS Firewall. This change allows Firewall Manager to configure Amazon Route 53 Resolver DNS Firewall associations. This permits you to use Firewall Manager to provide DNS Firewall protections for your VPCs throughout your organization in AWS Organizations.	March 17, 2021
Firewall Manager started tracking changes	Firewall Manager started tracking changes for its AWS managed policies.	March 03, 2021

Firewall Manager required permissions for API actions

When you set up Access control (p. 308) and writing permissions policies that you can attach to an IAM identity (identity-based policies), use the information in this section as a guide. For each AWS Firewall Manager API operation, you need to know the actions for which to grant permissions, and the AWS resource for which you 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.

> **Note**
> To specify an action, use the `fms:` prefix followed by the API operation name (for example, `fms:CreatePolicy`).
> This topic only list actions that require explicit resource permissions.

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

To use the following Firewall Manager API actions, you need permissions on the resource: `arn:aws:fms:`*`region`*`:`*`account`*`:policy/`*`ID`*.

- DeletePolicy
- GetComplianceDetail
- GetPolicy
- GetProtectionStatus
- ListComplianceStatus
- PutPolicy

Additionally, to use the Firewall Manager API action `PutNotificationChannel`, the Amazon SNS topic that you specify must allow the Firewall Manager service linked role to publish messages to it. The following shows an example SNS topic permission setting:

```
{
  "Sid": "AWSFirewallManagerSNSPolicy",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::account ID:role/aws-service-role/fms.amazonaws.com/
AWSServiceRoleForFMS"
  },
  "Action": "sns:Publish",
  "Resource": "SNS topic ARN"
}
```

For more information about Firewall Manager actions and resources, see the AWS Identity and Access Management guide topic Actions Defined by AWS Firewall Manager

For the full list of the API actions available for Firewall Manager, see AWS Firewall Manager API Reference.

Using service-linked roles for Firewall Manager

AWS Firewall Manager uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to Firewall Manager. Service-linked roles are predefined by Firewall Manager and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up Firewall Manager easier because you don't have to manually add the necessary permissions. Firewall Manager defines the permissions of its service-linked roles, and unless defined otherwise, only Firewall Manager can assume its roles. The defined permissions include

the trust policy and the permissions policy. That permissions policy can't be attached to any other IAM entity.

You can delete a service-linked role only after first deleting the role's related resources. This protects your Firewall Manager resources because you can't inadvertently remove permission to access the resources.

For information about other services that support service-linked roles, see AWS Services That Work with IAM and look for the services that have **Yes** in the **Service-Linked Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-linked role permissions for Firewall Manager

Firewall Manager uses the service-linked role AWSServiceRoleForFMS.

AWS Firewall Manager uses this service-linked role to write logs to Amazon Kinesis Data Firehose. This role is used only if you enable logging in AWS Firewall Manager. For more information, see Logging web ACL traffic information (p. 107).

The AWSServiceRoleForFMS service-linked role trusts the service to assume the role `fms.amazonaws.com`.

The permissions policies of the role allows Firewall Manager to complete the following actions on the specified resources:

- Action: `firehose:PutRecord` and `firehose:PutRecordBatch` on Amazon Kinesis Data Firehose data stream resources with a name that starts with "aws-fms-logs-." For example, `aws-fms-logs-us-east-2-analytics`.

You must configure permissions to allow an IAM entity (such as a user, group, or role) to create, edit, or delete a service-linked role. For more information, see Service-Linked Role Permissions in the *IAM User Guide*.

Creating a service-linked role for Firewall Manager

You don't need to manually create a service-linked role. When you enable Firewall Manager logging on the AWS Management Console, or you make a `PutLoggingConfiguration` request in the Firewall Manager CLI or the Firewall Manager API, Firewall Manager creates the service-linked role for you.

You must have the `iam:CreateServiceLinkedRole` permission to enable logging.

If you delete this service-linked role, and then need to create it again, you can use the same process to recreate the role in your account. When you enable Firewall Manager logging, Firewall Manager creates the service-linked role for you again.

Editing a service-linked role for Firewall Manager

Firewall Manager doesn't allow you to edit the AWSServiceRoleForFMS service-linked role. After you create a service-linked role, you can't change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a Service-Linked Role in the *IAM User Guide*.

Deleting a service-linked role for Firewall Manager

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don’t have an unused entity that is not actively monitored or maintained. However, you must clean up the resources for your service-linked role before you can manually delete it.

Note
If the Firewall Manager service is using the role when you try to delete the resources, then the deletion might fail. If that happens, wait for a few minutes and try the operation again.

To delete the service-linked role using IAM

Use the IAM console, the IAM CLI, or the IAM API to delete the AWSServiceRoleForFMS service-linked role. For more information, see Deleting a Service-Linked Role in the *IAM User Guide*.

Supported Regions for Firewall Manager service-linked roles

Firewall Manager supports using service-linked roles in the following AWS Regions.

Region Name	Region Identity	Support in Firewall Manager
US East (N. Virginia)	`us-east-1`	Yes
US East (Ohio)	`us-east-2`	Yes
US West (N. California)	`us-west-1`	Yes
US West (Oregon)	`us-west-2`	Yes
Asia Pacific (Mumbai)	`ap-south-1`	Yes
Asia Pacific (Osaka)	`ap-northeast-3`	Yes
Asia Pacific (Seoul)	`ap-northeast-2`	Yes
Asia Pacific (Singapore)	`ap-southeast-1`	Yes
Asia Pacific (Sydney)	`ap-southeast-2`	Yes
Asia Pacific (Tokyo)	`ap-northeast-1`	Yes
Canada (Central)	`ca-central-1`	Yes
Europe (Frankfurt)	`eu-central-1`	Yes
Europe (Ireland)	`eu-west-1`	Yes
Europe (London)	`eu-west-2`	Yes
Europe (Paris)	`eu-west-3`	Yes
South America (São Paulo)	`sa-east-1`	Yes

Logging and monitoring in Firewall Manager

Monitoring is an important part of maintaining the reliability, availability, and performance of Firewall Manager and your AWS solutions. You should collect monitoring data from all parts of your AWS solution so that you can more easily debug a multi-point failure if one occurs. AWS provides several tools for monitoring your Firewall Manager resources and responding to potential events:

Amazon CloudWatch Alarms

Using CloudWatch alarms, you watch a single metric over a time period that you specify. If the metric exceeds a given threshold, CloudWatch sends a notification to an Amazon SNS topic or AWS Auto Scaling policy. For more information, see Monitoring with Amazon CloudWatch (p. 370).

AWS CloudTrail Logs

CloudTrail provides a record of actions taken by a user, role, or an AWS service in Firewall Manager. Using the information collected by CloudTrail, you can determine the request that was made to Firewall Manager, the IP address from which the request was made, who made the request, when it was made, and additional details. For more information, see Logging API calls with AWS CloudTrail (p. 376).

Compliance validation for Firewall Manager

Third-party auditors assess the security and compliance of AWS services as part of multiple AWS compliance programs, such as SOC, PCI, FedRAMP, and HIPAA.

To learn whether Firewall Manager or other AWS services are in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying baseline environments on AWS that are security and compliance focused.
- Architecting for HIPAA Security and Compliance Whitepaper – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.

  Note
  Not all services are compliant with HIPAA.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- Evaluating Resources with Rules in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.
- AWS Audit Manager – This AWS service helps you continuously audit your AWS usage to simplify how you manage risk and compliance with regulations and industry standards.

Resilience in Firewall Manager

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

Infrastructure security in AWS Firewall Manager

As a managed service, AWS Firewall Manager is protected by the AWS global network security procedures that are described in the Amazon Web Services: Overview of Security Processes whitepaper.

You use AWS published API calls to access Firewall Manager through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

AWS Firewall Manager quotas

AWS Firewall Manager is subject to the following quotas (formerly referred to as limits).

AWS Firewall Manager has default quotas that you might be able to increase and fixed quotas.

The security group policies managed by Firewall Manager are subject to standard Amazon VPC quotas. For more information, see Amazon VPC Quotas in the Amazon VPC User Guide.

Each Firewall Manager Network Firewall policy creates a Network Firewall firewall with an associated firewall policy and its rule groups. These Network Firewall resources are subject to the quotas listed at AWS Network Firewall quotas in the *Network Firewall Developer Guide*.

Mutable quotas

AWS Firewall Manager has default quotas on the number of entities per Region. You can request an increase in these quotas.

All policy types

Resource	Default quota per Region
Accounts per organization in AWS Organizations	Varies. An invitation sent to an account counts against this quota. The count is returned if the invited account declines, the management account cancels the invitation, or the invitation expires.
Firewall Manager policies per organization in AWS Organizations	20. The Region specifications `Global` and `US East (N. Virginia)` refer to the same Region, so this limit applies to the total combined policies for the two of them.
Organizational units in scope per Firewall Manager policy	20
Accounts in scope of a Firewall Manager policy if you explicitly include and exclude individual accounts	200
Accounts in scope of a Firewall Manager policy if you do not explicitly include or exclude individual accounts	2,500
Tags that include or exclude resources per Firewall Manager policy	8

Common security group policies

Resource	Default quota per Region
Primary security groups per policy	1
Amazon VPC instances in scope per policy per account, including shared VPCs	100

Content audit security group policies

Resource	Default quota per Region
Audit security groups per policy	1
Applications per application list	50
Custom managed application lists for any setting in a policy	1
Custom managed application lists per account	10
Protocols per protocol list	5
Custom managed protocol lists for any setting in a policy	1
Custom managed protocol lists per account	10

AWS WAF policies

Resource	Default quota per Region
AWS WAF rule groups per Firewall Manager administrator account	100
AWS WAF Classic rule groups per Firewall Manager administrator account	10
Rule groups per AWS WAF policy	50
Total web ACL capacity units (WCU) for the rule groups in an AWS WAF policy	1500

DNS Firewall policies

Resource	Default quota per Region
DNS Firewall rule groups per Firewall Manager policy	2

Immutable quotas

The following per-Region quotas related to AWS Firewall Manager can't be changed.

Network Firewall policies

Resource	Quota per Region
Number of VPCs that can be automatically remediated for a single policy.	1,000
The number of IPV4 CIDRs that you can provide for a single policy.	50

Security group content audit policies

Resource	Quota per Region
Firewall Manager managed application lists for any setting in a policy	1
Firewall Manager managed protocol lists for any setting in a policy	1

AWS WAF Classic policies

Resource	Quota per Region
AWS WAF Classic rule groups per policy	2: 1 customer-created rule group and 1 AWS Marketplace rule group
AWS WAF Classic rules per Firewall Manager AWS WAF Classic rule group	10

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 resources.

You can add Shield Advanced protection for any of the following resource types:

- Amazon CloudFront distributions
- Amazon Route 53 hosted zones
- AWS Global Accelerator accelerators
- Application Load Balancers
- Elastic Load Balancing (ELB) load balancers
- Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP addresses

> **Protecting Network Load Balancers**
> You can't directly attach a Shield Advanced protection to a Network Load Balancer (NLB), but you can protect a Network Load Balancer by first associating an Amazon EC2 Elastic IP address to it and then adding the Elastic IP as a Shield Advanced protected resource. Some scaling tools, like AWS Elastic Beanstalk, don't let you automatically attach an Elastic IP to a Network Load Balancer. For those cases, you need to first associate the Elastic IP to the Network Load Balancer and then manually add the Shield Advanced protections to the Elastic IP.

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

All AWS customers benefit from the automatic protections of AWS Shield Standard, at no additional charge. AWS Shield Standard defends against the most common, frequently occurring network and transport layer DDoS attacks that target your website or applications. While AWS Shield Standard helps protect all AWS customers, you get particular benefit if you are using Amazon CloudFront and Amazon Route 53. These services receive comprehensive availability protection against all known infrastructure (Layer 3 and 4) attacks.

AWS Shield Advanced

For higher levels of protection against attacks, you can subscribe to AWS Shield Advanced. When you subscribe to AWS Shield Advanced and add specific resources to be protected, AWS Shield Advanced provides expanded DDoS attack protection for web applications running on the resources.

> **Note**
> AWS Shield Advanced only protects resources that you have specified either in Shield Advanced or through a AWS Firewall Manager Shield Advanced policy. It doesn't automatically protect your resources.

You can add Shield Advanced protection for any of the following resource types:

- Amazon CloudFront distributions
- Amazon Route 53 hosted zones
- AWS Global Accelerator accelerators
- Application Load Balancers
- Elastic Load Balancing (ELB) load balancers
- Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP addresses

> **Protecting Network Load Balancers**
> You can't directly attach a Shield Advanced protection to a Network Load Balancer (NLB), but you can protect a Network Load Balancer by first associating an Amazon EC2 Elastic IP address to it and then adding the Elastic IP as a Shield Advanced protected resource. Some scaling tools, like AWS Elastic Beanstalk, don't let you automatically attach an Elastic IP to a Network Load Balancer. For those cases, you need to first associate the Elastic IP to the Network Load Balancer and then manually add the Shield Advanced protections to the Elastic IP.

For example, if you use Shield Advanced to protect an Elastic IP address, Shield Advanced automatically deploys your network ACLs to the border of the AWS network during an attack. When your network ACLs are at the border of the network, Shield Advanced can provide protection against larger DDoS events. Typically, network ACLs are applied near your Amazon EC2 instances within your Amazon VPC. The network ACL can mitigate attacks only as large as your Amazon VPC and instance can handle. If the network interface attached to your Amazon EC2 instance can process up to 10 Gbps, volumes over 10 Gbps slow down and possibly block traffic to that instance. During an attack, Shield Advanced promotes your network ACL to the AWS border, which can process multiple terabytes of traffic. Your network ACL is able to provide protection for your resource well beyond your network's typical capacity. For more information about network ACLs, see Network ACLs.

The point at which Shield Advanced detects attacks and places mitigations depends on the architecture you use for your web applications. It varies based on characteristics like the type of instance you use, your instance size, and whether the instance type supports enhanced networking.

As an AWS Shield Advanced customer, you can contact the 24x7 AWS Shield Response Team (SRT) 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. With the assistance of the SRT, 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.

To use the services of the SRT, you must be subscribed to the Business Support plan or the Enterprise Support plan.

AWS Shield Advanced also offers some cost protection against spikes in your AWS bill that could result from a DDoS attack against your protected resources. For information, see Requesting a credit in AWS Shield Advanced (p. 352).

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.

When you add an AWS Shield Advanced protection to a resource, you can optionally include one or more additions to the protection. The protection additions vary by resource type and can include the following:

- A custom AWS WAF web ACL or rate-based rule, as described in Step 3: Configure layer 7 DDoS mitigation (p. 336).
- An Amazon Route 53 health check for health-based detection, as described in Shield Advanced health-based detection (p. 327).

Shield Advanced health-based detection

Shield Advanced health-based detection uses the health of your AWS resource to improve responsiveness and accuracy in attack detection and mitigation. To use health-based detection, you define the health check for your resource in Route 53 and then associate it with your Shield Advanced protection. For information about Route 53 health checks, see How Amazon Route 53 Checks the Health of Your Resources and Creating and Updating Health Checks.

> **Note**
> Do not use health checks with Route 53 hosted zones.

You can enable health-based detection for the following resource types:

- **Elastic IP addresses and Global Accelerator accelerators** – Health-based detection improves the accuracy of network-layer and transport-layer event detection and mitigation.

  When you protect an Elastic IP address or Global Accelerator accelerator with Shield Advanced, you reduce the threshold required to place a mitigation. Shield Advanced helps to provide quicker mitigation for attacks and mitigations for smaller attacks, even when traffic is within the application's capacity.

  When you add health-based detection, during periods when the associated Route 53 health check is unhealthy, Shield Advanced can place mitigations even more quickly and at lower thresholds.
- **CloudFront distributions and Application Load Balancers** – Health-based detection improves the accuracy of web request flood detection.

  When you protect a CloudFront distribution or Application Load Balancer with Shield Advanced, you receive web request flood detection alerts when there is a statistically significant deviation in traffic volume combined with significant changes in traffic self-similarity. Self-similarity is determined based on attributes like user agent, referrer, and URI.

  When you add health-based detection, you increase the likelihood that the alerts you receive are timely and actionable. With health-based detection, during periods when the associated Route 53 health check is unhealthy, Shield Advanced requires smaller deviations to alert and it reports events more quickly. When the associated Route 53 health check is healthy, Shield Advanced requires larger deviations to alert.

Shield Advanced proactive engagement

With proactive engagement, the Shield Response Team (SRT) engages with you directly if the Amazon Route 53 health check associated with your protected resource becomes unhealthy during an event that's detected by Shield Advanced. This allows you to engage with experts more quickly when the availability of your application might be affected by a suspected attack.

> **Note**
> To use proactive engagement for a protected resource, you must associate an Amazon Route 53 health check with the resource, as described in Shield Advanced health-based detection (p. 327).

Proactive engagement is available for network-layer and transport-layer events on Elastic IP addresses and AWS Global Accelerator accelerators, and for web request floods on Amazon CloudFront distributions and Application Load Balancers.

To use proactive engagement, you configure Shield Advanced health-based detection for a resource that you want the SRT to monitor. You then specify 1-10 contacts for proactive engagement. The SRT uses the information to contact you during a detected event that correlates with an unhealthy protected resource. After you provide your contact information, you can enable proactive engagement.

> **Note**
> To use proactive engagement, you must be subscribed to the Business Support plan or the Enterprise Support plan.

Shield Advanced protection groups

AWS Shield Advanced protection groups give you a self-service way to customize the scope of detection and mitigation by treating multiple protected resources as a single unit. Resource grouping can provide a number of benefits.

- Improve accuracy of detection.
- Reduce unactionable event notifications.
- Increase coverage of mitigation actions to include protected resources that also might be affected during an event.
- Accelerate time to mitigation of attacks with multiple similar targets.
- Facilitate automatic protection of newly created protected resources.

Protection groups can help reduce false positives in situations such as blue/green swap, where resources alternate between being near zero load and fully loaded. Another example is when you create and delete resources frequently while maintaining a load level that's shared among the members of the group. For situations such as these, monitoring individual resources can lead to false positives, while monitoring the health of the group of resources does not.

You can configure protection groups to include all protected resources, all resources of specific resource types, or individually specified resources. Newly protected resources that satisfy your protection group criteria are automatically included in your protection group. A protected resource can belong to multiple protection groups.

For information about your options and how to manage protection groups, see Managing AWS Shield Advanced protection groups (p. 344).

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 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.

The AWS Shield Response Team (SRT)

With AWS Shield Advanced, complex DDoS events can be escalated to the AWS Shield Response Team (SRT), 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.

You can also contact the SRT before or during a possible attack to develop and deploy custom mitigations. For example, if you are running a web application and only need ports 80 and 443 open, you can work with the SRT to pre-configure an ACL to only "Allow" ports 80 and 443.

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 SRT is not involved. You can, however, engage the SRT for guidance on implementing best practices such as AWS WAF common protections.
- **Engage the SRT:** 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 SRT, 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 SRT helps you triage the DDoS attack to identify attack signatures and patterns. With your consent, the SRT 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 SRT might proactively contact you. The SRT triages the DDoS event and creates AWS WAF mitigations. The SRT then contacts you for consent to apply the AWS WAF rules.

Important
The SRT can help you to analyze suspicious activity and assist you to mitigate the issue. This mitigation often requires the SRT 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 5: Configure AWS SRT support (p. 337) to proactively provide the SRT with the required permissions. Providing permission ahead of time helps prevent any delays in the event of an actual attack.
To use the services of the SRT, you must be subscribed to the Business Support plan or the Enterprise Support plan.

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.

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 SRT 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

Feature	AWS Shield Standard	AWS Shield Advanced
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, including automatic deployment of network ACLs to the AWS border during an attack		Yes
Custom application layer (layer 7) mitigations	Yes, through user-created AWS WAF ACLs. Incurs standard AWS WAF charges.	Yes, through user-created or SRT-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
Layer 7 attack notification	Yes, through AWS WAF. Incurs standard AWS WAF charges.	Yes

Feature	AWS Shield Standard	AWS Shield Advanced
Layer 7 attack forensics reports (Top talkers report, sampled requests, and more)	Yes, through AWS WAF web ACLs that you create. Incurs standard AWS WAF charges.	Yes, through AWS WAF web ACLs that you create or that the SRT creates on your behalf. AWS WAF is included with your Shield Advanced subscription.
Layer 3/4/7 attack historical report		Yes
Shield Response Team (SRT) Support (Must be subscribed to the Business Support plan or the Enterprise Support plan.)		
Event management during high severity events		Yes
Custom mitigations during attacks		Yes
Post-attack analysis		Yes
Cost Protection (Service credits for DDoS scaling charges)		
Elastic Load Balancing (ELB) load balancers		Yes
Amazon EC2 Elastic IP addresses		Yes
Amazon CloudFront distributions		Yes
Amazon Route 53 hosted zones		Yes
AWS Global Accelerator accelerators		Yes

AWS Shield Advanced benefits, including DDoS cost protection, are subject to your fulfillment of the 1-year subscription commitment.

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, AWS Firewall Manager, and AWS Shield Advanced (p. 368) and Logging API calls with AWS CloudTrail (p. 376).

Example AWS Shield Advanced use cases

You can use Shield Advanced to protect your resources in many types of scenarios. However, in some cases you should use other services or combine other services with Shield Advanced to offer the best protection. Following are examples of how to use Shield Advanced or other AWS services to help protect your resources.

Goal	Suggested services	Related service documentation
Protect a web application and RESTful APIs against a DDoS attack	Shield Advanced protecting an Amazon CloudFront distribution and an Application Load Balancer	Amazon Elastic Load Balancing Documentation, Amazon CloudFront Documentation
Protect a TCP-based application against a DDoS attack	Shield Advanced protecting a Network Load Balancer attached to an Elastic IP address	Amazon Elastic Load Balancing Documentation
Protect a UDP-based game server against a DDoS attack	Shield Advanced protecting an Amazon EC2 instance attached to an Elastic IP address	Amazon Elastic Compute Cloud Documentation

AWS Shield 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. The additional cost for AWS Shield Advanced includes basic AWS WAF protections for the resources that you protect with AWS Shield Advanced. Basic AWS WAF protections include your web ACLs, the rule groups that you manage, and any AWS Managed Rules that are provided free of charge with AWS WAF. AWS Shield Advanced costs do not cover additional paid features for AWS WAF, such as the use of the AWS WAF Bot Control rule group or AWS Marketplace rule groups.

AWS Shield Advanced charges do not increase with attack volume. This provides a predictable cost for your 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.

However, AWS Channel Resellers will pay a separate monthly fee for each member account. AWS Channel Resellers who resell AWS Shield Advanced to customers with more than one member account may contact us for additional billing support. With respect to such AWS Channel Resellers, AWS reserves the right to modify the monthly fee for AWS Shield Advanced. For more information, see the AWS Shield Advanced Pricing page.

Getting started with AWS Shield Advanced

This tutorial shows you how to get started with AWS Shield Advanced. For best results, perform the following steps in sequence.

Topics

Step 1: Subscribe to 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**
> Activate Shield Advanced for each AWS account that you want to protect. If you want to activate Shield Advanced for multiple accounts, we recommend that you use AWS Firewall Manager if you can. Firewall Manager doesn't support Amazon Route 53 or AWS Global Accelerator, but it supports the other resource types. For more information, see Getting started with AWS Firewall Manager AWS Shield Advanced policies (p. 256).

> **Note**
> The console guidance provided here is for the latest version of the AWS Shield console, released in 2020. In the console, you can switch between versions.

To subscribe to AWS Shield Advanced

1. Sign in to the AWS Management Console and open the AWS WAF & Shield console at https://console.aws.amazon.com/wafv2/.
2. In the **AWS Shield** navigation bar, choose **Getting started**. Choose **Subscribe to Shield Advanced**.
3. In the **Subscribe to Shield Advanced** page, read each term of the agreement, and then select all of the check boxes to indicate that you accept the terms.

   > **Important**
   > By choosing **Subscribe to Shield Advanced**, you subscribe to Shield Advanced and activate the service. To unsubscribe, you must contact AWS Support.

   Choose **Subscribe to Shield Advanced**.

Using AWS Shield Advanced with multiple accounts

You must activate Shield Advanced for each AWS account that you want to protect. To activate Shield Advanced for multiple accounts, we recommend that you use AWS Firewall Manager if you can. Firewall Manager doesn't support Amazon Route 53 or AWS Global Accelerator, but it supports the other resource types. For more information, see Getting started with AWS Firewall Manager AWS Shield Advanced policies (p. 256). Alternatively, for each account, log into the console and follow the procedure To subscribe to AWS Shield Advanced (p. 334).

If you activate Shield Advanced for multiple accounts that are in the same consolidated billing account family, the monthly subscription fee covers all those accounts. You don't pay extra subscription fees for individual accounts. You must own all the AWS accounts and resources in the account.

> **Note**
> AWS Channel resellers pay a separate monthly fee for each member account. AWS Channel resellers who resell AWS Shield Advanced to customers with more than one member account

can contact us for additional billing support. With respect to such AWS Channel resellers, AWS reserves the right to modify the monthly fee for AWS Shield Advanced. For more information, see the AWS Shield Advanced Pricing page.

The first time that you activate Shield Advanced from an account, you are presented with a pricing agreement. The pricing agreement is displayed on the console each time that you activate Shield Advanced from a different account. The pricing agreement covers all activated accounts in a consolidated billing family, but you must agree to the terms each time that you activate an account.

You can now go to Step 2: Add resources to protect (p. 335).

Step 2: Add resources to protect

After you subscribe to AWS Shield Advanced, as described in Step 1: Subscribe to AWS Shield Advanced (p. 334), you specify the resources that you want to protect.

If you are using AWS Firewall Manager to create a Firewall Manager Shield Advanced policy, you don't need to do this step. You already specified your resources in the Firewall Manager policy.

If you aren't using a Firewall Manager Shield Advanced policy, you can also specify resources later if you want, using the procedure at Adding AWS Shield Advanced protection to AWS resources (p. 342).

Note
Shield Advanced protects only resources that you have specified either in Shield Advanced or through a Firewall Manager Shield Advanced policy. It doesn't automatically protect your resources.

Note
The console guidance provided here is for the latest version of the AWS Shield console, released in 2020. In the console, you can switch between versions.

To choose the resources to protect with Shield Advanced

1. Do one of the following, depending on your starting point:
   - From the subscription confirmation page at the end of the procedure Step 1: Subscribe to AWS Shield Advanced (p. 334), choose **Add resources to protect**.
   - From the console navigation bar, choose **Protected Resources** and then choose **Add resources to protect**.
2. In the **Choose resources to protect with Shield Advanced** page, select the Regions and resource types that you want to protect, then choose **Load resources**.

   Note

   - If you want to protect an Amazon EC2 instance or a Network Load Balancer (NLB), you first must associate an Elastic IP address to it, and then choose the Elastic IP address as the resource to protect.
   - If you choose an Elastic IP address as the resource to protect, Shield Advanced protects whatever resource is associated with that Elastic IP address. Shield Advanced automatically identifies the type of resource that is associated with the Elastic IP address and applies the appropriate mitigations for that resource. This includes configuring network ACLs that are specific to the Elastic IP address. For more information about using Elastic IP addresses with your AWS resources, see the appropriate guide: Amazon Elastic Compute Cloud Documentation or Elastic Load Balancing Documentation.
   - Shield Advanced does not support EC2-Classic.
   - Some scaling tools, like AWS Elastic Beanstalk, do not let you automatically attach an Elastic IP to a Network Load Balancer. For those cases, you need to manually associate the Elastic IP.

3. Select the resources that you want to protect, then choose **Protect with Shield Advanced**.

You can now go to Step 3: Configure layer 7 DDoS mitigation (p. 336).

Step 3: Configure layer 7 DDoS mitigation

We recommend that you add web ACLs with rate-based rules as part of your AWS Shield Advanced protections. These rules can alert you to sudden spikes in traffic that might indicate a potential DDoS event. A rate-based rule counts the requests that arrive from any individual address in any five-minute period. If the number of requests exceeds the limit that you define, the rule can trigger an action such as sending you a notification. For more information about rate-base rules in AWS WAF, see Rate-based rule statement (p. 68).

> **Note**
> A resource can only be associated with one web ACL at a time. If you want to change web ACLs for a resource, remove the current web ACL association, and then associate the new web ACL. For more information, see Associating or disassociating a web ACL with an AWS resource (p. 26).

To configure layer 7 DDoS mitigation for a Region

Shield Advanced gives you the option to configure layer 7 DDoS mitigation for each Region where your chosen resources are located. Perform the following procedure for each one.

1. In the **Configure layer 7 DDoS mitigation** page, for each resource that isn't associated with a web ACL, either choose an existing web ACL or create a new web ACL.

   To create a web ACL, follow these steps:

   a. Choose **Create web ACL**.
   b. Enter a name. You can't change the name after you create the web ACL.
   c. Choose **Create**.
2. For each associated web ACL that doesn't have a rate-based rule defined, you can add one by choosing **Add rate limit rule** and then performing the following steps:

   a. Enter a name.
   b. Enter a rate limit. This is the maximum number of requests allowed in any five-minute period from any single IP address before the rate-based rule action is applied to the IP address. When the requests from the IP address fall below the limit, the action is discontinued.
   c. Set the rule action to count or block requests from IP addresses while their request counts are over the limit. The application and removal of the rule action might take effect a minute or two after the IP address request rate changes.
   d. Choose **Add rule**.
3. Choose **Next**.

To continue without adding web ACLs or rate-based rules

- Choose **Next**.

  > **Important**
  > If you use Shield Advanced within an AWS Firewall Manager Shield Advanced policy, you can't add a web ACL or rate-based rule. For all other resources, we recommend that, at a minimum, you attach a web ACL to each resource, even if that web ACL doesn't contain any rules.

You can now go to Step 4: Review and configure your settings (p. 337).

Step 4: Review and configure your settings

To review and configure your settings

1. When you finish adding your web ACLs and rate-based rules in the procedure Step 3: Configure layer 7 DDoS mitigation (p. 336), the console wizard proceeds to the **Configure health check based DDoS detection - *optional*** page. Health check based DDoS detection can help improve responsiveness to events. For information about this option, see Configure health check based DDoS detection (p. 346).

   For this tutorial, you won't configure health check based DDoS detection.

   Choose **Next**.
2. The console wizard proceeds to the **Create alarms and notifications - *optional*** page. Alarms and notifications allow you to be notified for events on your resources. For information about this option, see Create alarms and notifications (p. 347).

   For this tutorial, you won't create any alarms or notifications.

   Choose **Next**.
3. In the **Review and configure DDoS mitigation and visibility** page, review your settings. To make modifications, choose **Edit** in the area that you want to modify. This takes you back to the associated page in the console wizard. Make your changes, then choose **Next** in the subsequent pages until you return to the **Review and configure DDoS mitigation and visibility** page.
4. Choose **Finish configuration**.

The **Protected resources** page lists your newly protected resources.

You can now go to Step 5: Configure AWS SRT support (p. 337).

Step 5: Configure AWS SRT support

With AWS Shield Advanced, you can engage with the AWS Shield Response Team (SRT) if your application is unhealthy because of a possible DDoS attack.

> **Note**
> To contact the SRT, you must be subscribed to the Business Support plan or the Enterprise Support plan. If you are not subscribed to either plan, some of the options described in this section might not be visible in your account or accessible via the AWS Shield Advanced API.

You can contact the Shield Response Team (SRT) in one of the following ways:

- **Support case** – You can open a case under **AWS Shield** in the AWS Support Center. If your application is unhealthy, open a case using the highest severity available for your support plan and select either the **Phone** or **Chat** contact options. In the description for your case, provide as much detail as possible. Be sure to provide information about any protected resources that you think might be affected, and the current state of your end-user experience. For example, if your user experience is degraded or parts of your application are currently unavailable, provide that information.
- **Proactive engagement** – With AWS Shield Advanced proactive engagement, the SRT contacts you directly if the Amazon Route 53 health check associated with your protected resource becomes unhealthy during an event that is detected by Shield Advanced. For more information about this option, see Shield Advanced proactive engagement (p. 328).

Grant the SRT limited access to certain APIs and Amazon S3 buckets that you designate

AWS Shield Advanced automatically mitigates DDoS attacks against your resources. Shield Advanced detects web application layer vectors, like web request floods and low-and-slow bad bots, but does not automatically mitigate them. To mitigate web application layer vectors, you must employ AWS WAF rules or the SRT must employ the rules on your behalf.

> **Note**
> Shield Advanced detects web application layer events when you protect Amazon CloudFront distributions and Application Load Balancers. These events indicate a statistically significant deviation in traffic, compared to your application's historical baseline. You can choose to take no action if a deviation is expected or has not affected the health of your resource.

The SRT can assist you with web application layer events if you grant limited access to your Shield Advanced and AWS WAF APIs. You can revoke access at any time. The SRT engineers only access your APIs with your authorization, limited to the scope of your support engagement.

(Optional) Grant SRT access to an Amazon S3 bucket

To mitigate application layer events you can share additional log data, such as Application Load Balancer access logs, Amazon CloudFront logs, or logs from third party sources. For the SRT to view or process your logs, they must be in Amazon S3 buckets that satisfy the following requirements:

To authorize the SRT to assist with web application layer events on your behalf

1.
   - The buckets must be managed in one of the following ways:
     - (Option) The buckets are in the same AWS account as the web ACL.
     - (Option) The buckets are located in a separate account, and you have ensured that the SRT can access them. If you have multiple accounts in your organization, you can use a central Amazon S3 bucket in any of your Shield Advanced accounts within the organization. Alternately, you can use an Amazon S3 bucket in an account that does not have Shield Advanced providing it stores logs for an AWS WAF web ACL that is associated with a Shield Advanced protected resource.
     - (Option) The buckets are the storage destination for logging that is managed by AWS Firewall Manager for an AWS WAF policy.
   - The buckets can be either plaintext or SSE-S3 encrypted. For more information about Amazon S3 SSE-S3 encryption, see Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys (SSE-S3) in the Amazon Simple Storage Service Amazon Simple Storage Service Developer Guide.

     The SRT cannot view or process logs that are stored in buckets that are encrypted with keys stored in AWS Key Management Service (AWS KMS).
   - Shield Advanced allows you to give the SRT permission to access up to 10 buckets. If you want to give permission to more than 10, you need to edit the bucket policies manually.
2. In the AWS Shield console **Overview** page, under **Configure AWS SRT support**, choose **Edit SRT access**.
3. For the **SRT access setting**, select one of the following:

   - (Option) **Create a new role for the SRT to access my account** – For this option, Shield creates the role and automatically configures it for use. The new role allows the SRT to access your AWS Shield Advanced and AWS WAF resources. It also trusts the service principal `drt.shield.amazonaws.com`, which represents the SRT.
   - (Option) **Choose an existing role for the SRT to access my account** – For this option, you must modify the configuration of the role in AWS Identity and Access Management (IAM) as follows:
     - Attach the managed policy `AWSShieldDRTAccessPolicy` to the role. The `AWSShieldDRTAccessPolicy` managed policy gives the SRT access to your AWS Shield Advanced and AWS WAF resources. For more information, see Attaching and Detaching IAM Policies.

- Modify the role to trust the service principal `drt.shield.amazonaws.com`. This is the service principal that represents the SRT. For more information, see IAM JSON Policy Elements: Principal.

4. For each Amazon S3 bucket where your data or logs are stored, enter the name of the bucket and choose **Add Bucket**. You can add up to 10 buckets.

   This grants the SRT the following permissions on the bucket: `s3:GetBucketLocation`, `s3:GetObject`, and `s3:ListBucket`.

   If you want to give the SRT permission to access more than 10 buckets, you can do this by editing the additional bucket policies manually.

5. Choose **Save**.

To enable SRT proactive engagement

Shield Advanced proactive engagement allows you to engage with the SRT more quickly when the availability of your application is affected, because of a possible attack. When you have proactive engagement enabled, the SRT contacts you when a Shield Advanced event correlates to an unhealthy Route 53 health check on one or more of your protected resources.

1. In the AWS Shield console **Overview** page, under **Proactive engagement and contacts**, in the contacts area, choose **Edit**.

   In the **Edit contacts** page, provide the contact information for the people that you want the SRT to reach out to for proactive engagement.

   Note
   If you provide more than one contact, in the **Notes**, indicate the circumstances under which each contact should be used. Include primary and secondary contact designations, and provide the hours of availability and time zones for each contact.

2. Choose **Save**.

   The **Overview** page reflects the updated contact information.

3. Choose **Edit proactive engagement feature**, choose **Enable**, and then choose **Save**.

You can change SRT access and permissions at any time in the **Overview** page.

Continue to Step 6: Create a DDoS Dashboard in CloudWatch and Set CloudWatch Alarms (p. 339).

Step 6: Create a DDoS Dashboard in CloudWatch and Set CloudWatch Alarms

You can monitor potential DDoS activity using CloudWatch, which collects and processes raw data from Shield Advanced into readable, near real-time metrics. You can use statistics in Amazon CloudWatch to gain a perspective on how your web application or service is performing. For more information, see What is CloudWatch in the *Amazon CloudWatch User Guide*.

For instructions for creating a CloudWatch dashboard, see Monitoring with Amazon CloudWatch (p. 370). For information about specific Shield Advanced metrics that you can add to your dashboard, see AWS Shield Advanced metrics and alarms (p. 373).

As a final step for getting started with Shield Advanced, review the global threat dashboard, as described in Step 7: Monitor the global threat dashboard (p. 340).

Step 7: Monitor the global threat dashboard

The global threat dashboard provides a near real-time summary of the global AWS threat landscape. The threat landscape includes the largest attack, the top attack vectors, and the relative number of significant attacks. To view the history of significant DDoS attacks, you can customize the dashboard for different time durations. For more information, see Monitoring threats across AWS (p. 349).

Configuring AWS Shield Advanced setup

You can change your AWS Shield Advanced setup, for example to modify Shield Response Team (SRT) access to your account or adding or removing emergency contact information.

> **Note**
> To use the services of the Shield Response Team (SRT), you must be subscribed to the Business Support plan or the Enterprise Support plan.

> **Note**
> The console guidance provided here is for the latest version of the AWS Shield console, released in 2020. In the console, you can switch between versions.

With AWS Shield Advanced, you can engage with the AWS Shield Response Team (SRT) if your application is unhealthy because of a possible DDoS attack.

> **Note**
> To contact the SRT, you must be subscribed to the Business Support plan or the Enterprise Support plan. If you are not subscribed to either plan, some of the options described in this section might not be visible in your account or accessible via the AWS Shield Advanced API.

You can contact the Shield Response Team (SRT) in one of the following ways:

- **Support case** – You can open a case under **AWS Shield** in the AWS Support Center. If your application is unhealthy, open a case using the highest severity available for your support plan and select either the **Phone** or **Chat** contact options. In the description for your case, provide as much detail as possible. Be sure to provide information about any protected resources that you think might be affected, and the current state of your end-user experience. For example, if your user experience is degraded or parts of your application are currently unavailable, provide that information.
- **Proactive engagement** – With AWS Shield Advanced proactive engagement, the SRT contacts you directly if the Amazon Route 53 health check associated with your protected resource becomes unhealthy during an event that is detected by Shield Advanced. For more information about this option, see Shield Advanced proactive engagement (p. 328).

Grant the SRT limited access to certain APIs and Amazon S3 buckets that you designate

AWS Shield Advanced automatically mitigates DDoS attacks against your resources. Shield Advanced detects web application layer vectors, like web request floods and low-and-slow bad bots, but does not automatically mitigate them. To mitigate web application layer vectors, you must employ AWS WAF rules or the SRT must employ the rules on your behalf.

> **Note**
> Shield Advanced detects web application layer events when you protect Amazon CloudFront distributions and Application Load Balancers. These events indicate a statistically significant deviation in traffic, compared to your application's historical baseline. You can choose to take no action if a deviation is expected or has not affected the health of your resource.

The SRT can assist you with web application layer events if you grant limited access to your Shield Advanced and AWS WAF APIs. You can revoke access at any time. The SRT engineers only access your APIs with your authorization, limited to the scope of your support engagement.

(Optional) Grant SRT access to an Amazon S3 bucket

To mitigate application layer events you can share additional log data, such as Application Load Balancer access logs, Amazon CloudFront logs, or logs from third party sources. For the SRT to view or process your logs, they must be in Amazon S3 buckets that satisfy the following requirements:

To authorize the SRT to assist with web application layer events on your behalf

1.
   - The buckets must be managed in one of the following ways:
     - (Option) The buckets are in the same AWS account as the web ACL.
     - (Option) The buckets are located in a separate account, and you have ensured that the SRT can access them. If you have multiple accounts in your organization, you can use a central Amazon S3 bucket in any of your Shield Advanced accounts within the organization. Alternately, you can use an Amazon S3 bucket in an account that does not have Shield Advanced providing it stores logs for an AWS WAF web ACL that is associated with a Shield Advanced protected resource.
     - (Option) The buckets are the storage destination for logging that is managed by AWS Firewall Manager for an AWS WAF policy.
   - The buckets can be either plaintext or SSE-S3 encrypted. For more information about Amazon S3 SSE-S3 encryption, see Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys (SSE-S3) in the Amazon Simple Storage Service Amazon Simple Storage Service Developer Guide.

     The SRT cannot view or process logs that are stored in buckets that are encrypted with keys stored in AWS Key Management Service (AWS KMS).
   - Shield Advanced allows you to give the SRT permission to access up to 10 buckets. If you want to give permission to more than 10, you need to edit the bucket policies manually.
2. In the AWS Shield console **Overview** page, under **Configure AWS SRT support**, choose **Edit SRT access**.
3. For the **SRT access setting**, select one of the following:

   - (Option) **Create a new role for the SRT to access my account** – For this option, Shield creates the role and automatically configures it for use. The new role allows the SRT to access your AWS Shield Advanced and AWS WAF resources. It also trusts the service principal `drt.shield.amazonaws.com`, which represents the SRT.
   - (Option) **Choose an existing role for the SRT to access my account** – For this option, you must modify the configuration of the role in AWS Identity and Access Management (IAM) as follows:
     - Attach the managed policy `AWSShieldDRTAccessPolicy` to the role. The `AWSShieldDRTAccessPolicy` managed policy gives the SRT access to your AWS Shield Advanced and AWS WAF resources. For more information, see Attaching and Detaching IAM Policies.
     - Modify the role to trust the service principal `drt.shield.amazonaws.com`. This is the service principal that represents the SRT. For more information, see IAM JSON Policy Elements: Principal.
4. For each Amazon S3 bucket where your data or logs are stored, enter the name of the bucket and choose **Add Bucket**. You can add up to 10 buckets.

   This grants the SRT the following permissions on the bucket: `s3:GetBucketLocation`, `s3:GetObject`, and `s3:ListBucket`.

   If you want to give the SRT permission to access more than 10 buckets, you can do this by editing the additional bucket policies manually.
5. Choose **Save**.

To enable SRT proactive engagement

Shield Advanced proactive engagement allows you to engage with the SRT more quickly when the availability of your application is affected, because of a possible attack. When you have proactive engagement enabled, the SRT contacts you when a Shield Advanced event correlates to an unhealthy Route 53 health check on one or more of your protected resources.

1. In the AWS Shield console **Overview** page, under **Proactive engagement and contacts**, in the contacts area, choose **Edit**.

   In the **Edit contacts** page, provide the contact information for the people that you want the SRT to reach out to for proactive engagement.

   Note
   If you provide more than one contact, in the **Notes**, indicate the circumstances under which each contact should be used. Include primary and secondary contact designations, and provide the hours of availability and time zones for each contact.

2. Choose **Save**.

   The **Overview** page reflects the updated contact information.

3. Choose **Edit proactive engagement feature**, choose **Enable**, and then choose **Save**.

Managing resource protections in AWS Shield Advanced

You can add and configure AWS Shield Advanced protections for your resources. You can manage protections for a single resource and you can group your protected resources into logical collections for better event management. You can also track changes to your Shield Advanced protections using AWS Config.

Note
To use the services of the Shield Response Team (SRT), you must be subscribed to the Business Support plan or the Enterprise Support plan.

Note
The console guidance provided here is for the latest version of the AWS Shield console, released in 2020. In the console, you can switch between versions.

Topics

Adding AWS Shield Advanced protection to AWS resources

As part of enabling Shield Advanced for an account, you optionally choose initial resources to protect. You can add protection to more resources at any time.

Note
Shield Advanced protects only resources that you have specified either in Shield Advanced or through an AWS Firewall Manager Shield Advanced policy. It doesn't automatically protect your resources.

Shield Advanced offers advanced monitoring and protection for the following:

- Elastic Load Balancing (ELB) load balancers
- Amazon EC2 Elastic IP addresses
- Amazon CloudFront distributions
- Amazon Route 53 hosted zones
- AWS Global Accelerator accelerators

You can monitor and protect up to 1,000 resources for each of these resource types per AWS account. For example, you could protect 1,000 IP addresses, 1,000 distributions, and 1,000 load balancers in a single account. If you want to increase the number of resources that you can protect, contact the AWS Support Center.

If you add resources after your initial setup, you typically must add Shield Advanced protection for each resource. However, if you're using an AWS Firewall Manager Shield Advanced policy, you might not need to add resources yourself. If a new resource is within the Firewall Manager policy scope, Firewall Manager automatically includes it within the Shield Advanced policy protection.

Important
Before you perform the following procedure, you must complete the procedure in Step 1: Subscribe to AWS Shield Advanced (p. 334).

Note
The console guidance provided here is for the latest version of the AWS Shield console, released in 2020. In the console, you can switch between versions.

To add protection for an AWS resource

1. Sign in to the AWS Management Console and open the AWS WAF & Shield console at https://console.aws.amazon.com/wafv2/.
2. In the navigation pane, under AWS Shield choose **Protected resources**.
3. Choose **Add resources to protect**.
4. In the **Choose resources to protect with Shield Advanced** page, select the Regions and resource types that you want to protect, then choose **Load resources**.

   Note

   - If you want to protect an Amazon EC2 instance or a Network Load Balancer (NLB), you first must associate an Elastic IP address to it, and then choose the Elastic IP address as the resource to protect.
   - If you choose an Elastic IP address as the resource to protect, Shield Advanced protects whatever resource is associated with that Elastic IP address. Shield Advanced automatically identifies the type of resource that is associated with the Elastic IP address and applies the appropriate mitigations for that resource. This includes configuring network ACLs that are specific to the Elastic IP address. For more information about using Elastic IP addresses with your AWS resources, see the appropriate guide: Amazon Elastic Compute Cloud Documentation or Elastic Load Balancing Documentation.
   - Shield Advanced does not support EC2-Classic.
   - Some scaling tools, like AWS Elastic Beanstalk, do not let you automatically attach an Elastic IP to a Network Load Balancer. For those cases, you need to manually associate the Elastic IP.

5. Select the resources that you want to protect, then choose **Protect with Shield Advanced**.
6. Walk through the options and provide the protection settings that you want to apply to the resource. The options for adding protection to a resource are the same as for managing existing protections. See Configuring AWS Shield Advanced protections (p. 345).

Managing AWS Shield Advanced protection groups

Use protection groups to create logical collections of your protected resources and manage their protections as a group. For information about managing resource protections, see Configuring AWS Shield Advanced protections (p. 345). For information about protection groups, see Shield Advanced protection groups (p. 328).

Note
The console guidance provided here is for the latest version of the AWS Shield console, released in 2020. In the console, you can switch between versions.

Creating a Shield Advanced protection group

To create a protection group

1. Sign in to the AWS Management Console and open the AWS WAF & Shield console at https://console.aws.amazon.com/wafv2/.
2. In the AWS Shield navigation pane, choose **Protected resources**.
3. Choose the **Protection groups** tab, then choose **Create protection group**.
4. In the **Create protection group** page, provide a name for your group. You'll use this name to identify the group in your list of protected resources. You can't change the name of a protection group after you create it.
5. For **Protection grouping criteria**, select the criteria that you want Shield Advanced to use to identify the protected resources to include in the group. Make your additional selections based on the criteria that you've chosen.
6. For **Aggregation**, select how you want Shield Advanced to combine resource data for the group in order to detect, mitigate, and report events.

   - **Sum** – Use the total traffic across the group. This is a good choice for most cases. Examples include Elastic IP addresses for Amazon EC2 instances that scale manually or automatically.
   - **Mean** – Use the average of the traffic across the group. This is a good choice for resources that share traffic uniformly. Examples include accelerators and load balancers.
   - **Max** – Use the highest traffic from each resource. This is useful for resources that don't share traffic, and for resources that share traffic in a non-uniform way. Examples include Amazon CloudFront distributions and origin resources for CloudFront distributions.
7. Choose **Save** to save your protection group and return to the **Protected resources** page.

In the **Shield Events** page, you can view events for your protection group and drill down to see additional information for the protected resources that are in the group.

Updating a Shield Advanced protection group

To update a protection group

1. Sign in to the AWS Management Console and open the AWS WAF & Shield console at https://console.aws.amazon.com/wafv2/.
2. In the AWS Shield navigation pane, choose **Protected resources**.

3. In the **Protection groups** tab, select the check box next to the protection group that you want to modify.
4. In the protection group's page, choose **Edit**. Make your changes to the protection group settings.
5. Choose **Save** to save your changes.

Deleting a Shield Advanced protection group

To delete a protection group

1. Sign in to the AWS Management Console and open the AWS WAF & Shield console at https://console.aws.amazon.com/wafv2/.
2. In the AWS Shield navigation pane, choose **Protected resources**.
3. In the **Protection groups** tab, select the check box next to the protection group that you want to remove.
4. In the protection group's page, choose **Delete** and confirm the action.

Configuring AWS Shield Advanced protections

You can change the settings for your AWS Shield Advanced protections at any time. To do this, walk through the options for your selected protections and modify the settings that you need to change.

> **Note**
> The console guidance provided here is for the latest version of the AWS Shield console, released in 2020. In the console, you can switch between versions.

To manage protected resources

1. Sign in to the AWS Management Console and open the AWS WAF & Shield console at https://console.aws.amazon.com/wafv2/.
2. In the AWS Shield navigation pane, choose **Protected resources**.
3. In the **Protected resources** tab, select the resources that you want to protect.
4. Choose **Configure protections** and the resource specification option that you want.
5. Walk through each of the resource protection options, making changes as needed. The topics that follow cover each of the options.

Configure layer 7 DDoS mitigation

For protection against attacks on Amazon CloudFront and Application Load Balancer resources, you can add AWS WAF web ACLs and rate-based rules. For information about how AWS WAF works, see AWS WAF (p. 6).

If you use Shield Advanced within an AWS Firewall Manager Shield Advanced policy, you can't add a web ACL or rate-based rule.

To add web ACLs and rules

1. In the **Configure layer 7 DDoS mitigation** page, if the resource isn't already associated with a web ACL, you can choose an existing web ACL or create your own.

   To create a web ACL, follow these steps:

   a. Choose **Create web ACL**.
   b. Enter a name. You can't change the name after you create the web ACL.

c. Choose **Create**.

Note
If a resource is already associated with a web ACL, you can't change to a different web ACL. If you want to change the web ACL, you must first remove the associated web ACLs from the resource. For more information, see Associating or disassociating a web ACL with an AWS resource (p. 26).

To create your own web ACL, follow these steps:

2. For each associated web ACL that doesn't have a rate-based rule defined, you can add one by choosing **Add rate limit rule** and then performing the following steps:

   a. Enter a name.
   b. Enter a rate limit. This is the maximum number of requests allowed in any five-minute period from any single IP address before the rate-based rule action is applied to the IP address. When the requests from the IP address fall below the limit, the action is discontinued.
   c. Set the rule action to count or block requests from IP addresses while their request counts are over the limit. The application and removal of the rule action might take effect a minute or two after the IP address request rate changes.
   d. Choose **Add rule**.

3. Choose **Next**.

Configure health check based DDoS detection

You can use your Amazon Route 53 health checks to improve how AWS Shield Advanced detects and mitigates network-layer, transport-layer, and application layer attacks. For information about how this works, see Shield Advanced health-based detection (p. 327).

To get started with health-based detection, create a Route 53 health check for the AWS resource that you want to protect with Shield Advanced. For information about Route 53 health checks, see How Amazon Route 53 Checks the Health of Your Resources and Creating and Updating Health Checks. After you create your health check, associate it with your protection using the following procedure.

Note
You are responsible for ensuring that the health check you use is relevant to the health of the protected resource and that it remains available for use by the protection. Shield Advanced doesn't manage the health check in any way.

The following procedure shows how to associate an Amazon Route 53 health check with a protection.

To configure health-based DDoS detection

1. In the protections page **Configure health check based DDoS detection - *optional***, for the resource that you want to manage, under **Associated Health Check**, choose the ID of the health check that you want to associate with the protection.

   Note
   If you don't see the health check you need, go to the Route 53 console and verify the health check and its ID. For information, see Creating and Updating Health Checks.

2. Choose **Next**.

Shield Advanced health check status settings

The status of the health check that you associate with a protection can have the following values in the Shield console:

- **Healthy** — The health check is available and is reporting healthy.
- **Unhealthy** — The health check is available and is reporting unhealthy.
- **Unavailable** — The health check is not available for use by Shield Advanced. To resolve this, first disassociate the health check from the protection in Shield Advanced. Then, in Route 53, create a new health check for the protection and note its ID. Finally, associate the new health check with the protection following the procedure in this topic. Don't try to reassociate a health check that has been unavailable.

Create alarms and notifications

The following procedure shows how to manage CloudWatch alarms for protected resources.

> **Note**
> CloudWatch incurs additional costs. For CloudWatch pricing, see Amazon CloudWatch Pricing.

To create alarms and notifications

1. In the protections page **Create alarms and notifications - *optional***, configure the SNS topics for the alarms and notifications that you want to receive. For resources that you don't want notifications for, choose **No topic**. You can add an Amazon SNS topic or create a new topic.
2. To create an Amazon SNS topic, follow these steps:
   a. In the dropdown list, choose **Create an SNS topic**.
   b. Enter a topic name.
   c. Optionally enter an email address that the Amazon SNS messages will be sent to, and then choose **Add email**. You can enter more than one.
   d. Choose **Create**.
3. Choose **Next**.

Removing AWS Shield Advanced protection from an AWS resource

You can remove AWS Shield Advanced protection from any of your AWS resources at any time.

> **Important**
> Deleting an AWS resource doesn't remove the resource from AWS Shield Advanced. You must also remove the protection on the resource from AWS Shield Advanced, as described in this procedure.

> **Note**
> The console guidance provided here is for the latest version of the AWS Shield console, released in 2020. In the console, you can switch between versions.

Remove AWS Shield Advanced protection from an AWS resource

1. Sign in to the AWS Management Console and open the AWS WAF & Shield console at https://console.aws.amazon.com/wafv2/.
2. Choose **Protected resources**.
3. Select the resources whose protections you want to remove.
4. Choose **Delete protections**.
   - If you have an Amazon CloudWatch alarm configured for a protection, you are given the option to delete the alarm along with the protection. If you choose not to delete the alarm at this point, you can instead delete it later using the CloudWatch console.

Note
For protections that have an Amazon Route 53 health check configured, if you add the protection again later, the protection still includes the health check.

The preceding steps remove AWS Shield Advanced protection from specific AWS resources. They don't cancel your AWS Shield Advanced subscription. You will continue to be charged for the service. For information about your AWS Shield Advanced subscription, contact the AWS Support Center.

Removing a CloudWatch alarm from your Shield Advanced protections

To remove a CloudWatch alarm from your Shield Advanced protections, do one of the following:

- Delete the protection as described in Removing AWS Shield Advanced protection from an AWS resource (p. 347). Be sure to select the check box next to **Also delete related DDoSDetection alarm**.
- Delete the alarm using the CloudWatch console. The name of the alarm to delete starts with **DDoSDetectedAlarmForProtection**.

Tracking resource protection changes in AWS Config

You can record changes to the AWS Shield Advanced protection of your resources using AWS Config. You can then use this information to maintain a configuration change history for audit and troubleshooting purposes.

To record protection changes, enable AWS Config for each resource that you want to track. For more information, see Getting Started with AWS Config in the *AWS Config Developer Guide*.

You must enable AWS Config for each AWS Region that contains the tracked resources. You can enable AWS Config manually, or you can use the AWS CloudFormation template "Enable AWS Config" at AWS CloudFormation StackSets Sample Templates in the *AWS CloudFormation User Guide*.

If you enable AWS Config, you're charged as detailed on the AWS Config Pricing page.

Note
If you already have AWS Config enabled for the necessary Regions and resources, you don't need to do anything. AWS Config logs regarding protection changes to your resources start populating automatically.

After enabling AWS Config, use the US East (N. Virginia) Region in the AWS Config console to view the configuration change history for AWS Shield Advanced global resources.

View the change history for AWS Shield Advanced regional resources via the AWS Config console in the US East (N. Virginia), US East (Ohio), US West (Oregon), US West (N. California), Europe (Ireland), Europe (Frankfurt), Asia Pacific (Tokyo), and Asia Pacific (Sydney) Regions.

Responding to DDoS events

AWS automatically addresses layer 3 and layer 4 DDoS attacks. If you use Shield Advanced to protect your Amazon EC2 instances, during an attack Shield Advanced automatically deploys your Amazon VPC network ACLs to the border of the AWS network. This allows Shield Advanced to provide protection against larger DDoS events. For more information about network ACLs, see Network ACLs.

If DDoS alarms in CloudWatch indicate a possible layer 7 attack, you have the following options:

- Investigate and mitigate the attack on your own. If you determine that the 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. 22).

  If you use AWS Firewall Manager, you can add these rules to a Firewall Manager AWS WAF policy.
- If you're an AWS Shield Advanced customer, you also have the option of contacting the AWS Support Center to get help with mitigations. Critical and urgent cases are routed directly to DDoS experts. With AWS Shield Advanced, complex cases can be escalated to the AWS Shield Response Team (SRT), which has deep experience in protecting AWS, Amazon.com, and its subsidiaries. For more information about the SRT, see The AWS Shield Response Team (SRT) (p. 329).

  To get Shield Response Team (SRT) support, contact the AWS Support Center. Select the following options:
  - Case type: Technical Support
  - Service: Distributed Denial of Service (DDoS)
  - Category: Inbound to AWS
  - Severity: *Choose an appropriate option*

  When discussing with our representative, explain that you're an AWS Shield Advanced customer experiencing a possible DDoS 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 SRT can help you analyze the suspicious activity, and then assist you to mitigate the issue. This mitigation often requires the SRT 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 5: Configure AWS SRT support (p. 337) to proactively provide the SRT with the needed permissions. Providing permission ahead of time helps to prevent any delays in the event of an actual attack.

  You can also contact the SRT before or during a possible attack to develop and deploy custom mitigations. For example, if you're running a web application and need only ports 80 and 443 open, you can work with the SRT to preconfigure a web ACL to "allow" only ports 80 and 443.

  You authorize and contact the SRT at the account level. That is, if you use Shield Advanced within a Firewall Manager Shield Advanced policy, the account owner, not the Firewall Manager administrator, must contact the SRT for support. The Firewall Manager administrator can contact the SRT only for accounts that they own.

Reviewing DDoS events

Monitoring threats across AWS

Use the AWS Shield global threat dashboard to view trends and metrics about the DDoS threat landscape across Amazon EC2, Amazon CloudFront, Elastic Load Balancing, and Amazon Route 53.

The global threat dashboard provides a near real-time summary of the global AWS threat landscape, including the largest event, the top event vectors, and the relative number of significant events. You can customize the dashboard view for different time durations to see the history of significant DDoS events.

Note
The console guidance provided here is for the latest version of the AWS Shield console, released in 2020. In the console, you can switch between versions.

To view the global threat dashboard

1. Sign in to the AWS Management Console and open the AWS WAF & Shield console at https://console.aws.amazon.com/wafv2/.
2. In the **AWS Shield** navigation bar, choose **Global threat dashboard**.
3. Choose a time period.

You can use the information on the global threat dashboard to better understand the threat landscape and help you make decisions to better protect your AWS resources.

Shield Advanced metrics and reports

AWS Shield Advanced provides real-time metrics and reports for extensive visibility into events on your AWS resources.

These metrics and reports are available only for AWS Shield Advanced customers. To activate AWS Shield Advanced, see To subscribe to AWS Shield Advanced (p. 334).

You can view near real-time metrics about events, including the following:

- Event type
- Start time
- Duration
- Blocked bits or packets per second
- Passed bits or packets per second
- Source and destination IP addresses
- Protocol
- Resource ASN
- TCP flag

Details are available for active events and for past events that have occurred in the last 12 months.

Additionally, AWS Shield Advanced gives you insight into your overall traffic at the time of an event. You can review details about the most prevalent:

- IP addresses
- URLs
- Referrers
- ASNs
- Countries
- User Agents

Use this information to create AWS WAF rules to help prevent future attacks. For example, if you see that you have a lot of requests coming from a country that you don't typically do business in, you can create an AWS WAF rule to block requests from that country.

In addition to the metrics and reports described here, you can see a year's summary of events for all of your resources, regardless of their protection status, in the **Overview** page under **Events summary in past year**.

Note
The console guidance provided here is for the latest version of the AWS Shield console, released in 2020. In the console, you can switch between versions.

To review DDoS events

1. Sign in to the AWS Management Console and open the AWS WAF & Shield console at https://console.aws.amazon.com/wafv2/.
2. In the **AWS Shield** navigation bar, choose **Events**.

The **Events** page provides the following possible current status for events:

Mitigation in progress

Indicates a possible layer 3 or 4 event has been identified and AWS is attempting to address the issue.

Mitigated

Indicates a possible layer 3 or 4 event was identified. AWS responded to the event and it appears to be over.

Identified (ongoing)

Indicates a possible layer 7 event has been identified. AWS cannot address layer 7 events, so you must design your own layer mitigation processes. For more information on responding to possible layer 7 events, see Responding to DDoS events (p. 348).

Identified (subsided)

Indicates a possible layer 7 event was identified and appears to be over.

If you determine a possible event is underway, you can contact the Shield Response Team (SRT) through the AWS Support Center, or attempt to mitigate the event on your own by creating a new web access control list (web ACL).

The **Events** page also contains tabs with additional details for detection and mitigation of events and for top contributors during an event.

Detection and mitigation

The **Detection and mitigation** tab displays graphs that show the traffic that AWS Shield evaluated prior to reporting this event and the effect of any mitigation that was placed. It also provides metrics for infrastructure-layer Distributed Denial of Service (DDoS) events for resources in your own Amazon Virtual Private Cloud VPC and AWS Global Accelerator accelerators. Detection and mitigation metrics are not included for Amazon CloudFront or Amazon Route 53 resources.

AWS Shield evaluates traffic to your protected resource along multiple dimensions. When an anomaly is detected, Shield creates an event and reports the traffic dimension where the anomaly was observed. If the protected resource is an Elastic IP address, Classic Load Balancer (CLB), Application Load Balancer, or Global Accelerator accelerator, Shield automatically creates a mitigation for the protected resource. This protects your resource from receiving excess traffic and from receiving traffic that matches a known DDoS event signature.

You might notice a difference between the traffic volume shown in the detection graphs and the pass and drop metrics reported on the mitigation graphs. If the traffic that resulted in an event subsides before a mitigation is created, this traffic doesn't appear in mitigation metrics. The time difference between the first data point in detection metrics and the first data point in mitigation metrics is the

latency between detection and mitigation of an event. Detection metrics are based on sampled network flows while mitigation metrics are based on traffic that's observed by the mitigation systems. Mitigation metrics are a more precise measurement of the traffic into your resource. The SRT might leave a mitigation in place after the excess traffic has subsided. Shield might maintain long-running mitigations on resources that are frequently targeted.

Top contributors

The **Top contributors** tab provides insight into where traffic is coming from during a detected event. You can view the highest volume contributors, sorted by aspects such as protocol, source port, and TCP flags. You can download the information and you can adjust the units used and the number of contributors to display.

Top contributors provide additional metric dimensions that you can use to understand the network traffic that's sent to your resource during an event. These metrics are based on sampled network flows for both legitimate and potentially unwanted traffic. Some or all of the traffic that you see in the metrics might have been passed to your resource or blocked by Shield.

Top contributor data might be unavailable for a metric if no significant contributors can be identified. Events that are more likely to have metrics with top contributors are large volume events and events where the traffic sources aren't highly distributed.

Keep in mind that source attributes like source IP address and source ASN might be spoofed or reflected. Spoofed packets are common in DDoS events where an attacker wants to obfuscate the real source of the traffic. Reflected packets are common in DDoS events where an attacker wants to generate a larger, amplified flood of traffic to a target by reflecting the attack of services on the Internet that are usually legitimate. In these cases, the source IP address or source ASN are valid, but they are not the actual source of the attack.

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.

   Note
   You should always test your rules first by initially using `Count` rather than `Block`. After you're comfortable that the new rule is identifying the correct requests, you can modify your rule to block those requests.
4. Monitor those counts to determine if the source of the requests should be blocked. If the volume of requests continues to be unusually high, change your web ACL to block those requests.

   For more information, see Creating a web ACL (p. 22).

AWS provides preconfigured templates to get you started quickly. The templates include a set of AWS WAF rules that you can customize and use to block common web-based attacks. For more information, see AWS WAF Security Automations.

Requesting a credit in AWS Shield Advanced

If you're subscribed to AWS Shield Advanced and you experience a DDoS attack that increases utilization of an Shield Advanced protected resource, you can request a credit for charges related to the increased utilization to the extent that it is not mitigated by Shield Advanced. Credits are available only for increased charges that result directly from a DDoS attack in the following areas: Amazon CloudFront HTTP/HTTPS requests, CloudFront data transfer out, Amazon Route 53 queries, AWS Global Accelerator

data transfer, load balancer capacity units for Application Load Balancer, and usage spikes on protected Amazon Elastic Compute Cloud (Amazon EC2) instances.

To be eligible to receive a credit, before the attack began, you must have done the following for the resources for which you want a credit:

- Added Shield Advanced protection to the resources. Protected resources added during an attack are not eligible for cost protection. Enabling Shield Advanced on your AWS account does not automatically enable Shield Advanced protection for individual resources. For more information about how to protect AWS resources using Shield Advanced, see Adding AWS Shield Advanced protection to AWS resources (p. 342).
- Associated an AWS WAF web ACL with applicable CloudFront and Application Load Balancer protected resources. For more information about how to associate web ACLs with AWS resources, see Managing and using a web access control list (web ACL) (p. 17).
- Defined an AWS WAF rate-based rule in block mode for applicable CloudFront and Application Load Balancer protected resources. For more information about how to create AWS WAF rate-based rules, see Rate-based rule statement (p. 68).
- Implemented applicable best practices, according to the guidance at AWS Best Practices for DDoS Resiliency, to configure your application in a way that minimizes cost during a DDoS attack.

To apply for a credit, submit a billing case through the AWS Support Center. Be sure to include the following in your request:

- The words "DDoS Concession" in the subject line
- The dates and times of each event interruption for which you're requesting a credit
- The AWS services and specific resources that were affected

> **Important**
> To be eligible for a credit, you must submit your credit request within 15 days of the end of the billing month in which the attack occurred.

After you submit a request, the AWS Shield Response Team (SRT) will validate whether a DDoS attack occurred and, if so, whether any protected resources scaled to absorb the attack. If AWS determines that protected resources scaled to absorb the DDoS attack, AWS will issue a credit for that portion of traffic that AWS determines was caused by the attack. Credits are valid for 12 months.

Security in AWS Shield

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. The effectiveness of our security is regularly tested and verified by third-party auditors as part of the AWS compliance programs. To learn about the compliance programs that apply to Shield, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your organization's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using Shield. The following topics show you how to configure Shield to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Shield resources.

Topics

Data protection in Shield

The AWS shared responsibility model applies to data protection in AWS Shield. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. This content includes the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form fields such as a **Name** field. This includes when you work with Shield or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Shield entities—such as protections—are encrypted at rest, except in certain Regions where encryption is not available, including China (Beijing) and China (Ningxia). Unique encryption keys are used for each Region.

Identity and access management in AWS Shield

Access to AWS Shield requires credentials. Those credentials must have permissions to access AWS resources, such as an AWS Shield resource or an Amazon S3 bucket. The following sections provide

details on how you can use AWS Identity and Access Management (IAM) and Shield 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 first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.
- **IAM user** – An IAM user is an identity within your AWS account that has specific custom permissions (for example, permissions to create a rule in Shield). 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 AWS tools, you must sign the request yourself. Shield 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 an IAM identity that you can create in your account that has specific permissions. An IAM role is similar to an IAM user in that it is an AWS identity with permissions policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session. IAM roles with temporary credentials are useful in the following situations:

  - **Federated user access** – Instead of creating an IAM user, you can use existing 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*.

  - **AWS service access** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. 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 that are running on an EC2 instance and making AWS CLI or 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 that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances 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 Shield resources. For example, you must have permissions to create a Shield *protection* or list *attacks*.

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

- Overview of managing access permissions to your AWS Shield resources (p. 356)
- Using identity-based policies (IAM policies) for AWS Shield (p. 360)
- Shield required permissions for API actions (p. 363)

AWS Identity and Access Management

Shield 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 Shield 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 Shield 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). 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 Shield resources and operations

In AWS Shield, the resources are *protections* and *attacks*. These resources have unique Amazon Resource Names (ARNs) associated with them, as shown in the following table.

Name in AWS Shield Console	Name in AWS Shield SDK/CLI	ARN Format	
Event or attack	`AttackDetail`	`arn:aws:shield::account:attack/ID`	
Protection	`Protection`	`arn:aws:shield::account:protection/ID`	

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

```
arn:aws:shield::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 Shield resource.
- *ID*: The ID of the Shield 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 protections for the account `111122223333`:

```
arn:aws:shield::111122223333:protection/*
```

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

AWS Shield provides a set of operations to work with Shield 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 a Shield 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 a Shield resource to that user, the user can create a Shield resource. However, your AWS account, to which the user belongs, owns the Shield resource.

- If you create an IAM role in your AWS account with permissions to create a Shield resource, anyone who can assume the role can create a Shield resource. Your AWS account, to which the role belongs, owns the Shield resource.
- With AWS Shield, to create a protection or describe an attack associated with a specific resource, a user must have an access to the resource itself in addition to having access to the Shield resource. For example to create a protection for an Amazon CloudFront distribution, the user needs read access for the distribution to protect. To describe an attack against a CloudFront distribution, the user needs read access to the distribution.

Managing access to resources

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

> **Note**
> These sections discuss using IAM in the context of AWS Shield. 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 Shield 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 Shield 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*.

  AWS Shield allows cross-account resource access, but it doesn't allow you to create cross-account resource protections. You can only create protections for resources from within the account that owns those resources.

The following is an example policy that grants permissions for the `shield:ListProtections` action on all resources. Shield 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": "2016-06-02",
    "Statement": [
        {
            "Sid": "ListProtections",
            "Effect": "Allow",
            "Action": [
                "shield:ListProtections"
            ],
            "Resource": "*"
        }
    ]
}
```

For more information about using identity-based policies with Shield, see Using identity-based policies (IAM policies) for AWS Shield (p. 360). 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 Shield doesn't support resource-based policies.

Specifying policy elements: Actions, effects, resources, and principals

For each AWS Shield resource (see AWS Shield resources and operations (p. 357)), the service defines a set of API operations (see Shield required permissions for API actions (p. 363)). To grant permissions for these API operations, Shield 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 Shield resources and operations (p. 357).
- **Action** – You use action keywords to identify resource operations that you want to allow or deny. For example, the `shield:CreateRuleGroup` permission allows the user permissions to perform the AWS Shield `CreateRuleGroup` 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 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 Shield 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 Shield API actions and the resources that they apply to, see Shield required permissions for API actions (p. 363).

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 Shield. However, there are general AWScondition keys that you can use as appropriate. For a complete list of AWS keys, see Available Keys for Conditions in the *IAM User Guide*.

Using identity-based policies (IAM policies) for AWS Shield

This section 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 Shield 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 Shield resources. For more information, see Overview of managing access permissions to your AWS Shield resources (p. 356).

For a table that shows all the AWS Shield API actions and the resources that they apply to, see Shield required permissions for API actions (p. 363).

Topics

Permissions required to use the AWS Shield console

The AWS Shield console provides an integrated environment for you to create and manage Shield resources. The console provides many features and workflows that often require permissions to create an Shield resource in addition to the API-specific permissions that are documented in the Shield required permissions for API actions (p. 363). For more information about these additional console permissions, see Customer managed policy examples (p. 361).

AWS managed (predefined) policies for AWS Shield

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*.

AWS Shield uses the AWS managed policy `AWSShieldDRTAccessPolicy` that you can use to grant the Shield Response Team (SRT) access to your account. This allows the SRT to perform actions on your account, to manage your AWS WAF rules and Shield protections. To use this, you create a role and pass it to the Shield API operation, associate SRT role. In the API, this is `AssociateDRTRole`. In the CLI, it's `associate-drt-role`. For more information about this policy, see Step 5: Configure AWS SRT support (p. 337).

> **Note**
> You can review AWS managed permissions policies by signing in to the IAM console and searching for the policies.

You also can create your own custom IAM policies to allow permissions for AWS Shield 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 Shield 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 on the console. As you attach policies to the user, you can verify that the user can perform various operations on 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 Shield resource, see Creating IAM Roles in the *IAM User Guide*.

Example topics

- Example 1: Give users read-only access to Shield, CloudFront, and CloudWatch (p. 361)
- Example 2: Give users full access to Shield, CloudFront, and CloudWatch (p. 362)

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 Shield, CloudFront, and CloudWatch

The following policy grants users read-only access to Shield an associated resources, including Amazon CloudFront resources, and Amazon CloudWatch metrics. It's useful for users who need permission to view the settings in Shield protections and attacks and to monitor metrics in CloudWatch. These users can't create, update, or delete Shield resources.

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "ProtectedResourcesReadAccess",
                "Effect": "Allow",
                "Action": [
                    "cloudfront:List*",
                    "elasticloadbalancing:List*",
                    "route53:List*",
                    "cloudfront:Describe*",
                    "elasticloadbalancing:Describe*",
                    "route53:Describe*",
                    "cloudwatch:Describe*",
                    "cloudwatch:Get*",
                    "cloudwatch:List*",
                    "cloudfront:GetDistribution*",
                    "globalaccelerator:ListAccelerators",
```

```
                    "globalaccelerator:DescribeAccelerator"
                ],
                "Resource": [
                    "arn:aws:elasticloadbalancing:*:*:*",
                    "arn:aws:cloudfront::*:*",
                    "arn:aws:route53:::hostedzone/*",
                    "arn:aws:cloudwatch:*:*:*:*",
                    "arn:aws:globalaccelerator::*:*"
                ]
            },
            {
                "Sid": "ShieldReadOnly",
                "Effect": "Allow",
                "Action": [
                    "shield:List*",
                    "shield:Describe*",
                    "shield:Get*"
                ],
                "Resource": "*"
            }
    ]
}
```

Example 2: Give users full access to Shield, CloudFront, and CloudWatch

The following policy lets users perform any Shield 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 Shield administrators.

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "ProtectedResourcesReadAccess",
                "Effect": "Allow",
                "Action": [
                    "cloudfront:List*",
                    "elasticloadbalancing:List*",
                    "route53:List*",
                    "cloudfront:Describe*",
                    "elasticloadbalancing:Describe*",
                    "route53:Describe*",
                    "cloudwatch:Describe*",
                    "cloudwatch:Get*",
                    "cloudwatch:List*",
                    "cloudfront:GetDistribution*",
                    "globalaccelerator:ListAccelerators",
                    "globalaccelerator:DescribeAccelerator"
                ],
                "Resource": [
                    "arn:aws:elasticloadbalancing:*:*:*",
                    "arn:aws:cloudfront::*:*",
                    "arn:aws:route53:::hostedzone/*",
                    "arn:aws:cloudwatch:*:*:*:*",
                    "arn:aws:globalaccelerator::*:*"
                ]
            },
            {
                "Sid": "ShieldFullAccess",
                "Effect": "Allow",
                "Action": [
                    "shield:*"
                ],
                "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*.

Shield required permissions for API actions

When you set up Access control (p. 356) and writing permissions policies that you can attach to an IAM identity (identity-based policies), use the information in this section as a guide. For each AWS Shield API operation, you need to know the 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.

> **Note**
> To specify an action, use the `shield:` prefix followed by the API operation name (for example, `shield:CreateProtection`).
> The following list only includes actions that require explicit resource permissions.

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

For each action, we list the actions and the associated policy resource specifications.

AssociateDRTLogBucket

API Actions – `shield:AssociateDRTLogBucket`, `s3:GetBucketPolicy`, `s3:PutBucketPolicy`

Resource – `arn:aws:s3:::`*`bucket_name`*`/`*`optional_object_key`*

AssociateDrtRole

API Actions – `shield:AssociateDrtRole`, `iam:GetRole`, `iam:ListAttachedRolePolicies`, `iam:PassRole`

Resource – `arn:aws:iam::`*`account-id`*`:role/`*`role-id`*

CreateProtection

API Actions – `shield:CreateProtection`

Resource – `arn:aws:shield::`*`account`*`:protection/`*`ID`*

DeleteProtection

API Actions – `shield:DeleteProtection`

Resource – `arn:aws:shield::`*`account`*`:protection/`*`ID`*

DescribeAttack

API Actions – `shield:DescribeAttack`

Resource – `arn:aws:shield::`*`account`*`:attack/`*`ID`*

DescribeDrtAccess

API Actions – `shield:DescribeDrtAccess`, `s3:GetBucketPolicy`

Resource – `arn:aws:s3:::`*`bucket_name`*`/`*`optional_object_key`*

DescribeProtection

API Actions – `shield:DescribeProtection`

Resource – `arn:aws:shield::account:protection/ID`

DisassociateDRTLogBucket

API Actions – `shield:DisassociateDRTLogBucket`, `s3:DeleteBucketPolicy`, `s3:GetBucketPolicy`, `s3:PutBucketPolicy`

Resource – `arn:aws:s3:::bucket_name/optional_object_key`

For more information about Shield actions and resources, see the AWS Identity and Access Management guide topic Actions Defined by AWS Shield.

For a full list of the API actions available for Shield, see AWS Shield Advanced API Reference.

Logging and monitoring in Shield

Monitoring is an important part of maintaining the reliability, availability, and performance of Shield and your AWS solutions. You should collect monitoring data from all parts of your AWS solution so that you can more easily debug a multi-point failure if one occurs. AWS provides several tools for monitoring your Shield resources and responding to potential events:

Amazon CloudWatch Alarms

Using CloudWatch alarms, you watch a single metric over a time period that you specify. If the metric exceeds a given threshold, CloudWatch sends a notification to an Amazon SNS topic or AWS Auto Scaling policy. For more information, see Monitoring with Amazon CloudWatch (p. 370).

AWS CloudTrail Logs

CloudTrail provides a record of actions taken by a user, role, or an AWS service in Shield. Using the information collected by CloudTrail, you can determine the request that was made to Shield, the IP address from which the request was made, who made the request, when it was made, and additional details. For more information, see Logging API calls with AWS CloudTrail (p. 376).

Compliance validation for Shield

Third-party auditors assess the security and compliance of Shield as part of multiple AWS compliance programs, such as SOC, PCI, FedRAMP, HIPAA, and others.

For a list of AWS services in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using Shield is determined by the sensitivity of your data, your organization's compliance objectives, and applicable laws and regulations. If your use of Shield is subject to compliance with standards like HIPAA, PCI, or FedRAMP, AWS provides resources to help:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- Architecting for HIPAA Security and Compliance Whitepaper – This publication describes how companies can use AWS to create HIPAA-compliant applications.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- AWS Config – This AWS service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Well-Architected Framework – The AWS Well-Architected Framework helps you build secure cloud applications.

Resilience in Shield

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

Infrastructure security in AWS Shield

As a managed service, AWS Shield is protected by the AWS global network security procedures that are described in Amazon Web Services: Overview of Security Processes.

You use AWS published API calls to access Shield through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

AWS Shield Advanced quotas

AWS Shield Advanced is subject to the following default quotas (formerly referred to as limits). To increase these quotas, go to the AWS Support Center.

Resource	Default quota
Maximum number of protected resources for each resource type that AWS Shield Advanced offers protection for, per account.	1,000
Maximum number of protection groups, per account.	100
Maximum number of individual protected resources that you can specifically include in a protection group. In the API, this applies to the `Members` that you specify when you set the protection group `Pattern` to `ARBITRARY`. In the console, this applies to the resources that you select for the protection grouping **Choose from protected resources**.	1,000

Monitoring AWS WAF, AWS Firewall Manager, and AWS Shield Advanced

Monitoring is an important part of maintaining the reliability, availability, and performance of your services.

> **Note**
> For information about monitoring your Shield Advanced resources and identifying possible DDoS events using Shield Advanced, see AWS Shield (p. 325).

As you start monitoring these services, 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, Firewall Manager, Shield Advanced 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 Advanced. You can configure some of these tools to do the monitoring for you, while other 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 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 Amazon EC2 Auto Scaling policy. Alarms invoke actions for sustained state changes only. CloudWatch alarms will 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 CloudFront Activity Using CloudWatch.

  Note
  CloudWatch metrics and alarms are not enabled for Firewall Manager.

  Not only can you use CloudWatch to monitor AWS WAF and Shield Advanced metrics as described in Monitoring with Amazon CloudWatch (p. 370), you also should use CloudWatch to monitor activity for your Amazon CloudFront, Amazon API Gateway, Application Load Balancer, and AWS AppSync resources. For more information, see Monitoring CloudFront Activity Using CloudWatch in the *Amazon CloudFront Developer Guide*, Tracing, Logging, and Monitoring an API Gateway API, CloudWatch Metrics for Your Application Load Balancer, and Monitoring and Logging in the *AWS AppSync Developer Guide*.
- **Amazon CloudWatch Logs** – Monitor, store, and access your log files from AWS CloudTrail or other sources. For more information, see What is Amazon CloudWatch Logs?.
- **Amazon CloudWatch Events** – Automate your AWS services and respond automatically to system events. Events from AWS services are delivered to CloudWatch Events in near real time, and you can specify automated actions to take when an event matches a rule that you write. For more information, see What is Amazon CloudWatch Events?
- **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 API calls with AWS CloudTrail (p. 376) and Working with CloudTrail Log Files in the *AWS CloudTrail User Guide*.
- **AWS Config** – View the configuration of AWS resources in your AWS account, including how the resources are related to one another and how they were configured in the past so that you can see how the configurations and relationships change over time.

Manual monitoring tools

Another important part of monitoring AWS WAF and AWS Shield Advanced involves manually monitoring those items that the CloudWatch alarms don't cover. You can view the AWS WAF, Shield Advanced, CloudWatch, and other AWS Management 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.

- For example, to 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 web requests (p. 28).
- 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 Amazon CloudWatch, which collects and processes raw data from AWS WAF and AWS Shield Advanced into readable, near real-time metrics. You can use statistics in Amazon CloudWatch to gain a perspective on how your web application or service is performing. For more information, see What is CloudWatch in the *Amazon CloudWatch User Guide*.

Note
CloudWatch metrics and alarms are not enabled for Firewall Manager.

Creating Amazon CloudWatch alarms

You can create an Amazon 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 AWS Shield Advanced.

Note
Amazon CloudWatch metrics and alarms are not enabled for AWS Firewall Manager.

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, choose the Region where your AWS resources are located. For more information, see AWS service endpoints.

   To view AWS WAF metrics for CloudFront, you must choose the US East (N. Virginia) Region.
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/WAFV2, at a command prompt use the following command:

```
aws cloudwatch list-metrics --namespace "AWS/WAFV2"
```

  For Shield Advanced, at a command prompt use the following command:

```
aws cloudwatch list-metrics --namespace "AWS/DDoSProtection"
```

AWS WAF metrics and dimensions

The `WAF` namespace includes the following metrics and dimensions.

Web ACL, rule group, and rule metrics

Metric	Description
`AllowedRequests`	The number of allowed web requests. Reporting criteria: There is a nonzero value. Valid statistics: Sum
`BlockedRequests`	The number of blocked web requests. Reporting criteria: There is a nonzero value. Valid statistics: Sum
`CountedRequests`	The number of counted web requests. Reporting criteria: There is a nonzero value. A counted web request is one that matches at least one of the rules. Request counting is typically used for testing. Valid statistics: Sum
`PassedRequests`	The number of passed requests. This is only used for requests that go through a rule group evaluation without matching any of the rule group rules. Reporting criteria: There is a nonzero value. Passed requests are requests that don't match any of the rules in the rule group. Valid statistics: Sum

Web ACL, rule group, and rule dimensions

Dimension	Description
`Region`	Required for all protected resource types except for Amazon CloudFront distributions.
`Rule`	One of the following: • The metric name of the `Rule`. • ALL, which represents all rules within a WebACL or `RuleGroup`. • `Default_Action` (only when combined with the `WebACL` dimension), which represents the action assigned to any request that doesn't match any rule with either an allow or block action.
`RuleGroup`	The metric name of the `RuleGroup`.
`WebACL`	The metric name of the `WebACL`.

Label and AWS WAF Bot Control metrics

Metric	Description
`AllowedRequests`	The number of labels applied to web requests by rules that had an allow action setting. Reporting criteria: There is a nonzero value. Valid statistics: Sum
`BlockedRequests`	The number of labels applied to web requests by rules that had a block action setting. Reporting criteria: There is a nonzero value. Valid statistics: Sum
`CountedRequests`	The number of labels applied to web requests by rules that had a count action setting. Reporting criteria: There is a nonzero value. Valid statistics: Sum

Label and AWS WAF Bot Control dimensions

Dimension	Description
`Region`	Required for all protected resource types except for Amazon CloudFront distributions.
`WebACL`	The metric name of the `WebACL`.
`RuleGroup`	The metric name of the `RuleGroup`. Used for the metric `CountedRequests`.
`LabelNamespace`	The namespace prefix of the web ACL or rule group where the matching rule is specified. Used for the metrics `AllowedRequests` and `BlockedRequests`.
`Label`	The label applied to the web request by the matching rule. Used for the metrics `AllowedRequests` and `BlockedRequests`.

Free bot visibility metrics

Metric	Description
`SampleAllowedRequests`	The percentage of sampled requests that have allow action. Reporting criteria: There is a nonzero value. Valid statistics: Sum
`SampleBlockedRequests`	The percentage of sampled requests that have block action.

Metric	Description
	Reporting criteria: There is a nonzero value. Valid statistics: Sum

Free bot visibility dimensions

Dimension	Description
`Region`	Required for all protected resource types except for Amazon CloudFront distributions.
`WebACL`	The metric name of the `WebACL`.
`BotCategory`	The metric name of the of the detected bot category, based on the web request labels.

AWS Shield Advanced metrics and alarms

This section discusses the metrics and alarms available with AWS Shield Advanced.

AWS Shield Advanced metrics

Shield Advanced reports metrics to Amazon CloudWatch on an AWS resource more frequently during DDoS events than while no events are underway. Shield Advanced reports metrics once a minute during an event, and then once right after the event ends. While no events are underway, Shield Advanced reports metrics once a day, at a time assigned to the resource. This periodic report keeps the metrics active and available for use in custom CloudWatch alarms.

For the global services Amazon CloudFront and Amazon Route 53, metrics are reported in the US East (N. Virginia) Region.

Detection metrics

Shield Advanced provides the following detection metrics and dimensions.

Detection metrics

Metric	Description
`DDoSDetected`	Indicates whether a DDoS event is underway for a particular Amazon Resource Name (ARN). This metric has a value of 1 during an event and a value of 0 otherwise.
`DDoSAttackBitsPerSecond`	The number of bits observed during a DDoS event for a particular Amazon Resource Name (ARN). This metric is available only for layer 3 and layer 4 DDoS events. This metric has a non-zero value during an event and a value of 0 otherwise. Units: Bits

Metric	Description
DDoSAttackPacketsPerSecond	The number of packets observed during a DDoS event for a particular Amazon Resource Name (ARN). This metric is available only for layer 3 and layer 4 DDoS events. This metric has a non-zero value during an event and a value of 0 otherwise. Units: Packets
DDoSAttackRequestsPerSecond	The number of requests observed during a DDoS event for a particular Amazon Resource Name (ARN). This metric is available only for layer 7 DDoS events. The metric is reported only for the most significant layer 7 events. This metric has a non-zero value during an event and a value of 0 otherwise. Units: Requests

Shield Advanced posts the `DDoSDetected` metric with no other dimensions. The remaining detection metrics include the `AttackVector` dimensions that correspond to the type of attack, from the following list:

- `ACKFlood`
- `ChargenReflection`
- `DNSReflection`
- `GenericUDPReflection`
- `MemcachedReflection`
- `MSSQLReflection`
- `NetBIOSReflection`
- `NTPReflection`
- `PortMapper`
- `RequestFlood`
- `RIPReflection`
- `SNMPReflection`
- `SSDPReflection`
- `SYNFlood`
- `UDPFragment`
- `UDPTraffic`
- `UDPReflection`

Mitigation metrics

Shield Advanced provides the following mitigation metrics and dimensions.

Mitigation metrics

Metric	Description
`VolumePacketsPerSecond`	The number of packets per second that were dropped or passed by a mitigation that was deployed in response to a detected event. Units: packets

Mitigation dimensions

Dimension	Description
`ResourceArn`	Amazon Resource Name (ARN)
`MitigationAction`	The outcome of an applied mitigation. Possible values are `Pass` or `Drop`.

Top contributors metrics

Shield Advanced provides the following top contributors metrics and dimensions.

Top contributors metrics

Metric	Description
`VolumePacketsPerSecond`	The number of packets per second for a top contributor. Units: packets
`VolumeBitsPerSecond`	The number of bits per second for a top contributor. Units: bits

Shield Advanced posts top contributors metrics by dimension combinations that characterize the event contributors. You can use any of the following combinations of dimensions for any of the top contributors metrics:

- `ResourceArn`, `Protocol`
- `ResourceArn`, `Protocol`, `SourcePort`
- `ResourceArn`, `Protocol`, `DestinationPort`
- `ResourceArn`, `Protocol`, `SourceIp`
- `ResourceArn`, `Protocol`, `SourceAsn`
- `ResourceArn`, `TcpFlags`

Top contributors dimensions

Dimension	Description
`ResourceArn`	Amazon Resource Name (ARN).
`Protocol`	IP protocol name, either `TCP` or `UDP`.

Dimension	Description
SourcePort	Source TCP or UDP port.
DestinationPort	Destination TCP or UDP port.
SourceIp	Source IP address.
SourceAsn	Source autonomous system number (ASN).
TcpFlags	A combination of flags present in a TCP packet, separated by a dash (-). Monitored flags are ACK, FIN, RST, SYN. This dimension value always appears sorted alphabetically. For example, ACK-FIN-RST-SYN, ACK-SYN, and FIN-RST.

Creating AWS Shield Advanced alarms

You can use AWS Shield Advanced metrics for Amazon CloudWatch alarms. CloudWatch sends notifications or automatically make changes to the resources that you are monitoring based on rules that you define.

For detailed instructions on creating a CloudWatch alarm, see the Amazon CloudWatch User Guide. When creating the alarm on the CloudWatch console, to use the Shield Advanced metrics, after choosing **Create an alarm**, choose **AWSDDOSProtectionMetrics** . You can then create an alarm based on a specific volume of traffic, or you can trigger an alarm whenever a metric is non-zero. The second option triggers an alarm for any potential attack that Shield Advanced observes.

> **Note**
> The **AWSDDOSProtectionMetrics** are available only to Shield Advanced customers.

For more information, see What is CloudWatch in the *Amazon CloudWatch User Guide*.

AWS Firewall Manager notifications

AWS Firewall Manager doesn't record metrics, so you can't create Amazon CloudWatch alarms specifically for Firewall Manager. However, you can configure Amazon SNS notifications to alert you to potential attacks. To create Amazon SNS notifications in Firewall Manager, see To create an Amazon SNS topic in Firewall Manager (console) (p. 259).

Logging API calls with AWS CloudTrail

AWS WAF, AWS Shield Advanced, and AWS Firewall Manager are integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service. CloudTrail captures a subset of API calls for these services as events, including calls from the AWS WAF, Shield Advanced or Firewall Manager consoles and from code calls to the AWS WAF, Shield Advanced, or Firewall Manager APIs. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AWS WAF, Shield Advanced, or Firewall Manager. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to these services, the IP address that the request was made from, who made the request, when it was made, and additional details.

To learn more about CloudTrail, including how to configure and enable it, see the AWS CloudTrail User Guide.

CloudTrail is enabled on your AWS account when you create the account. When supported event activity occurs in AWS WAF, Shield Advanced, or Firewall Manager, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for AWS WAF, Shield Advanced, or Firewall Manager, create a trail. A trail enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail on the console, the trail applies to all Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for Creating a Trail
- CloudTrail Supported Services and Integrations
- Configuring Amazon SNS Notifications for CloudTrail
- Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts

AWS WAF information in AWS CloudTrail

All AWS WAF actions are logged by AWS 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 event or log entry contains information about who generated the request. The identity information 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.

Example: AWS WAF log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. AWS CloudTrail log files contain one or more log entries. An event 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. CloudTrail log files are not an ordered stack trace of the public API calls, so they do not appear in any specific order.

The following are examples of CloudTrail log entries for AWS WAF web ACL operations.

Example: CloudTrail log entry for `CreateWebACL`

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "principalId",
    "arn": "arn:aws:sts::112233445566:assumed-role/Admin",
    "accountId": "112233445566",
    "accessKeyId": "accessKeyId",
    "sessionContext": {
```

```
      "sessionIssuer": {
        "type": "Role",
        "principalId": "principalId",
        "arn": "arn:aws:iam::112233445566:role/Admin",
        "accountId": "112233445566",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2019-11-06T03:43:07Z"
      }
    }
  },
  "eventTime": "2019-11-06T03:44:21Z",
  "eventSource": "wafv2.amazonaws.com",
  "eventName": "CreateWebACL",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "10.0.0.1",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/78.0.3904.87 Safari/537.36",
  "requestParameters": {
    "name": "foo",
    "scope": "CLOUDFRONT",
    "defaultAction": {
      "block": {}
    },
    "description": "foo",
    "rules": [
      {
        "name": "foo",
        "priority": 1,
        "statement": {
          "geoMatchStatement": {
            "countryCodes": [
              "AF",
              "AF"
            ]
          }
        },
        "action": {
          "block": {}
        },
        "visibilityConfig": {
          "sampledRequestsEnabled": true,
          "cloudWatchMetricsEnabled": true,
          "metricName": "foo"
        }
      }
    ],
    "visibilityConfig": {
      "sampledRequestsEnabled": true,
      "cloudWatchMetricsEnabled": true,
      "metricName": "foo"
    }
  },
  "responseElements": {
    "summary": {
      "name": "foo",
      "id": "ebbcb976-8d59-4d20-8ca8-4ab2f6b7c07b",
      "description": "foo",
      "lockToken": "67551e73-49d8-4363-be48-244deea72ea9",
      "aRN": "arn:aws:wafv2:us-west-2:112233445566:global/webacl/foo/
ebbcb976-8d59-4d20-8ca8-4ab2f6b7c07b"
    }
  },
```

```
  "requestID": "c51521ba-3911-45ca-ba77-43aba50471ca",
  "eventID": "afd1a60a-7d84-417f-bc9c-7116cf029065",
  "eventType": "AwsApiCall",
  "apiVersion": "2019-04-23",
  "recipientAccountId": "112233445566"
}
```


Example: CloudTrail log entry for `GetWebACL`


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AssumedRole",
    "arn": "arn:aws:sts::112233445566:assumed-role/Admin/admin",
    "accountId": "112233445566",
    "accessKeyId": "accessKeyId",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AssumedRole",
        "arn": "arn:aws:iam::112233445566:role/Admin",
        "accountId": "112233445566",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2019-11-06T19:17:20Z"
      }
    }
  },
  "eventTime": "2019-11-06T19:18:28Z",
  "eventSource": "wafv2.amazonaws.com",
  "eventName": "GetWebACL",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "10.0.0.1",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/78.0.3904.87 Safari/537.36",
  "requestParameters": {
    "name": "foo",
    "scope": "CLOUDFRONT",
    "id": "webacl"
  },
  "responseElements": null,
  "requestID": "f2db4884-4eeb-490c-afe7-67cbb494ce3b",
  "eventID": "7d563cd6-4123-4082-8880-c2d1fda4d90b",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "apiVersion": "2019-04-23",
  "recipientAccountId": "112233445566"
}
```


Example: CloudTrail log entry for `UpdateWebACL`


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "principalId",
    "arn": "arn:aws:sts::112233445566:assumed-role/Admin",
    "accountId": "112233445566",
    "accessKeyId": "accessKeyId",
```

```
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "principalId",
        "arn": "arn:aws:iam::112233445566:role/Admin",
        "accountId": "112233445566",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2019-11-06T19:17:20Z"
      }
    }
  },
  "eventTime": "2019-11-06T19:20:56Z",
  "eventSource": "wafv2.amazonaws.com",
  "eventName": "UpdateWebACL",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "10.0.0.1",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/78.0.3904.87 Safari/537.36",
  "requestParameters": {
    "name": "foo",
    "scope": "CLOUDFRONT",
    "id": "ebbcb976-8d59-4d20-8ca8-4ab2f6b7c07b",
    "defaultAction": {
      "block": {}
    },
    "description": "foo",
    "rules": [
      {
        "name": "foo",
        "priority": 1,
        "statement": {
          "geoMatchStatement": {
            "countryCodes": [
              "AF"
            ]
          }
        },
        "action": {
          "block": {}
        },
        "visibilityConfig": {
          "sampledRequestsEnabled": true,
          "cloudWatchMetricsEnabled": true,
          "metricName": "foo"
        }
      }
    ],
    "visibilityConfig": {
      "sampledRequestsEnabled": true,
      "cloudWatchMetricsEnabled": true,
      "metricName": "foo"
    },
    "lockToken": "67551e73-49d8-4363-be48-244deea72ea9"
  },
  "responseElements": {
    "nextLockToken": "a6b54c01-7975-4e6d-b7d0-2653cb6e231d"
  },
  "requestID": "41c96e12-9790-46ab-b145-a230f358f2c2",
  "eventID": "517a10e6-4ca9-4828-af90-a5cff9756594",
  "eventType": "AwsApiCall",
  "apiVersion": "2019-04-23",
  "recipientAccountId": "112233445566"
```

```
}
```

Example: CloudTrail log entry for `DeleteWebACL`

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "principalId",
    "arn": "arn:aws:sts::112233445566:assumed-role/Admin/sheqiang-Isengard",
    "accountId": "112233445566",
    "accessKeyId": "accessKeyId",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "principalId",
        "arn": "arn:aws:iam::112233445566:role/Admin",
        "accountId": "112233445566",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2019-11-06T19:17:20Z"
      }
    }
  },
  "eventTime": "2019-11-06T19:25:17Z",
  "eventSource": "wafv2.amazonaws.com",
  "eventName": "DeleteWebACL",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "10.0.0.1",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/78.0.3904.87 Safari/537.36",
  "requestParameters": {
    "name": "foo",
    "scope": "CLOUDFRONT",
    "id": "ebbcb976-8d59-4d20-8ca8-4ab2f6b7c07b",
    "lockToken": "a6b54c01-7975-4e6d-b7d0-2653cb6e231d"
  },
  "responseElements": null,
  "requestID": "71703f89-e139-440c-96d4-9c77f4cd7565",
  "eventID": "2f976624-b6a5-4a09-a8d0-aa3e9f4e5187",
  "eventType": "AwsApiCall",
  "apiVersion": "2019-04-23",
  "recipientAccountId": "112233445566"
}
```

Example: AWS WAF classic log file entries

AWS WAF Classic is the prior version of AWS WAF. For information, see AWS WAF Classic (p. 138).

The log entry demonstrates the `CreateRule`, `GetRule`, `UpdateRule`, and `DeleteRule` operations:

```
{
  "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"
    }
  ]
}
```

AWS Shield Advanced information in CloudTrail

AWS Shield Advanced supports logging the following actions as events in CloudTrail log files:

- ListAttacks
- DescribeAttack
- CreateProtection
- DescribeProtection
- DeleteProtection
- ListProtections
- CreateSubscription
- DescribeSubscription
- GetSubscriptionState
- DeleteSubscription

Every event or log entry contains information about who generated the request. The identity information 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 the CloudTrail userIdentity Element.

Example: Shield Advanced log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event 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. CloudTrail log files 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 `DeleteProtection` and `ListProtections` actions.


```
[
 {
   "eventVersion": "1.05",
   "userIdentity": {
     "type": "IAMUser",
     "principalId": "123456789098765432123 1",
     "arn": "arn:aws:iam::123456789012:user/SampleUser",
     "accountId": "123456789012",
     "accessKeyId": "1AFGDT647FHU83JHFI81H",
     "userName": "SampleUser"
   },
   "eventTime": "2018-01-10T21:31:14Z",
   "eventSource": "shield.amazonaws.com",
   "eventName": "DeleteProtection",
   "awsRegion": "us-east-1",
   "sourceIPAddress": "AWS Internal",
   "userAgent": "aws-cli/1.14.10 Python/3.6.4 Darwin/16.7.0 botocore/1.8.14",
   "requestParameters": {
```

```
      "protectionId": "12345678-5104-46eb-bd03-agh4j8rh3b6n"
    },
    "responseElements": null,
    "requestID": "95bc0042-f64d-11e7-abd1-1babdc7aa857",
    "eventID": "85263bf4-17h4-43bb-b405-fh84jhd8urhg",
    "eventType": "AwsApiCall",
    "apiVersion": "AWSShield_20160616",
    "recipientAccountId": "123456789012"
  },
  {
    "eventVersion": "1.05",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "12345678909876543212 3",
      "arn": "arn:aws:iam::123456789012:user/SampleUser",
      "accountId": "123456789012",
      "accessKeyId": "1AFGDT647FHU83JHFI81H",
      "userName": "SampleUser"
    },
    "eventTime": "2018-01-10T21:30:03Z",
    "eventSource": "shield.amazonaws.com",
    "eventName": "ListProtections",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "aws-cli/1.14.10 Python/3.6.4 Darwin/16.7.0 botocore/1.8.14",
    "requestParameters": null,
    "responseElements": null,
    "requestID": "6accca40-f64d-11e7-abd1-1bjfi8urhj47",
    "eventID": "ac0570bd-8dbc-41ac-a2c2-987j90j3h78f",
    "eventType": "AwsApiCall",
    "apiVersion": "AWSShield_20160616",
    "recipientAccountId": "123456789012"
  }
]
```


AWS Firewall Manager information in CloudTrail

AWS Firewall Manager supports logging the following actions as events in CloudTrail log files:

- AssociateAdminAccount
- DeleteNotificationChannel
- DeletePolicy
- DisassociateAdminAccount
- PutNotificationChannel
- PutPolicy
- GetAdminAccount
- GetComplianceDetail
- GetNotificationChannel
- GetPolicy
- ListComplianceStatus
- ListPolicies

Every event or log entry contains information about who generated the request. The identity information 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 the CloudTrail userIdentity Element.

Example: Firewall Manager log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event 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. CloudTrail log files 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 `GetAdminAccount-->` action.


```
{
                "eventVersion": "1.05",
                "userIdentity": {
                                "type": "AssumedRole",
                                "principalId": "123456789098765432123l",
                                "arn": "arn:aws:sts::123456789012:assumed-role/Admin/
SampleUser",
                                "accountId": "123456789012",
                                "accessKeyId": "1AFGDT647FHU83JHFI81H",
                                "sessionContext": {
                                                "attributes": {
                                                                "mfaAuthenticated":
 "false",
                                                                "creationDate":
 "2018-04-14T02:51:50Z"
                                                               },
                                                "sessionIssuer": {
                                                                "type": "Role",
                                                                "principalId":
 "123456789098765432123l",
                                                                "arn":
 "arn:aws:iam::123456789012:role/Admin",
                                                                "accountId":
 "123456789012",
                                                                "userName": "Admin"
                                                                 }
                                                  }
                                },
                "eventTime": "2018-04-14T03:12:35Z",
                "eventSource": "fms.amazonaws.com",
                "eventName": "GetAdminAccount",
                "awsRegion": "us-east-1",
                "sourceIPAddress": "72.21.198.65",
                "userAgent": "console.amazonaws.com",
                "requestParameters": null,
                "responseElements": null,
                "requestID": "ae244f41-3f91-11e8-787b-dfaafef95fc1",
                "eventID": "5769af1e-14b1-4bd1-ba75-f023981d0a4a",
                "eventType": "AwsApiCall",
                "apiVersion": "2018-01-01",
                "recipientAccountId": "123456789012"
    }
```

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 (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. For information about endpoints, see AWS service 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. 390).

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
- 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).

Related information

The following related resources can help you as you work with this service.

The following resources are available for AWS WAF, AWS Shield Advanced, and AWS Firewall Manager.

- **Guidelines for Implementing AWS WAF** – Technical publication with current recommendations for implementing AWS WAF to protect existing and new web applications.
- **AWS discussion forums** – A community-based forum for discussing technical questions related to this and other AWS services.
- **Getting Started Resource Center** – Information to help you get started building on AWS.
- **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.

The following resources are available for Amazon Web Services.

- **Classes & Workshops** – Links to role-based and specialty courses, in addition to 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 webpage 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

update-history-change	update-history-description	update-history-date
Firewall Manager supports Network Firewall log filtering (p. 296)	AWS Firewall Manager now supports log filtering for Network Firewall policies.	October 4, 2021
Added regex match statement (p. 69)	You can now match web requests against a single regular expression.	September 22, 2021
Rate-based rules inside AWS WAF rule groups (p. 68)	You can now define rate-based rules inside AWS WAF rule groups. In AWS Firewall Manager, this capability is fully supported for AWS WAF policies.	September 13, 2021
Firewall Manager supports AWS WAF log filtering (p. 285)	AWS Firewall Manager now supports log filtering for AWS WAF policies.	August 31, 2021
Automatically remove out-of-scope resource protections in AWS Firewall Manager (p. 280)	AWS Firewall Manager allows you to automatically remove protections from resources that leave policy scope.	August 25, 2021
Added versioning to managed rule groups (p. 30)	Managed rule group providers can now version their rule groups.	August 9, 2021
Modify AWS Firewall Manager administrator requirements (p. 249)	You can use the organization's management account as the Firewall Manager administrator account. This had been disallowed.	August 2, 2021
Firewall Manager quota increase (p. 322)	Increased the number of Amazon VPC instances that you can have in scope of a Firewall Manager policy from 10 to 100.	July 28, 2021
AWS Firewall Manager support for AWS Network Firewall route table monitoring (p. 284)	AWS Firewall Manager now supports route table monitoring, and provides remediation action recommendations to security administrators for AWS Network Firewall policies with misconfigured routes.	July 8, 2021
AWS WAF additional text transformation options (p. 78)	There are additional text transformations that you can apply to web request components before inspecting requests.	June 24, 2021

Modified naming for Firewall Manager AWS WAF policy resources (p. 284)	The naming for the web ACLs, rule groups, and logging that Firewall Manager manages for your AWS WAF policies has changed.	May 26, 2021
Updated AWS Managed Rules for AWS WAF (p. 47)	AWS Managed Rules for AWS WAF added support for labeling to IP reputation lists and removed suffixes on rule names for Amazon IP reputation list.	May 4, 2021
Add support for AWS Organizations Delegated Administrator (p. 249)	When you set the AWS Firewall Manager administrator account, Firewall Manager now designates the account as the AWS Organizations delegated administrator for Firewall Manager. With this change, when you set the Firewall Manager administrator account, you must provide a member account other than the organization's management account. This change doesn't affect your existing settings.	April 30, 2021
Updated AWS Managed Rules for AWS WAF (p. 47)	AWS Managed Rules for AWS WAF added the AWS WAF Bot Control rule set.	April 1, 2021
Set individual rule actions to count in a rule group (p. 20)	You can now set the individual rule actions in a rule group to count. The information for the existing override, which is at the rule group level, has been corrected.	April 1, 2021
Scope-down statement for managed rule groups (p. 81)	You can now use a scope-down statement with managed rule groups in the same way as you can with a rate-based statement.	April 1, 2021
Log filtering (p. 107)	You can now filter the web ACL traffic that you log based on rule action and label.	April 1, 2021
AWS WAF labels on web requests (p. 54)	You can configure rules to add labels to matching web requests and to match on labels that are added by other rules.	April 1, 2021

AWS WAF Bot Control (p. 94)	You can monitor and control bot traffic with the new AWS WAF Bot Control feature, which combines a new Bot Control managed rule group with web request labeling, scope-down statements, and log filtering.	April 1, 2021
Firewall Manager supports Amazon Route 53 Resolver DNS Firewall policies (p. 298)	AWS Firewall Manager supports central management of Amazon Route 53 Resolver DNS Firewall outbound DNS traffic filtering for your VPCs.	March 31, 2021
Custom request and response handling (p. 89)	You can include custom headers in web requests that AWS WAF allows or counts and you can send custom responses for web requests that AWS WAF blocks. Available for web ACL default action and rule action settings.	March 29, 2021
Updated AWS Managed Rules for AWS WAF (p. 47)	AWS Managed Rules for AWS WAF updated the following rule groups: core rule set (CRS), admin protection, known bad inputs, and Linux operating system.	March 3, 2021
Inspect a web request body as parsed JSON (p. 77)	Added the option to inspect the web request body as parsed and filtered JSON. This is in addition to the existing option to inspect the web request body as plain text.	February 12, 2021
Expanded AWS Shield Advanced event details in console (p. 350)	Shield Advanced customers can access additional metrics and reports for events in the Shield console **Events** page.	January 22, 2021
Firewall Manager supports AWS Network Firewall policies (p. 293)	AWS Firewall Manager supports central management of AWS Network Firewall network traffic filtering for your VPCs.	November 17, 2020
Add support for AWS Shield Advanced protection groups (p. 328)	You can now group your protected resources into logical groups and manage their protections collectively.	November 13, 2020
Added support for AWS AppSync to AWS WAF (p. 6)	You can now associate a AWS WAF web ACL with your AWS AppSync GraphQL API. This change is only available in the latest version of AWS WAF and not in AWS WAF Classic.	October 1, 2020

Updated AWS Managed Rules for AWS WAF (p. 47)	AWS Managed Rules for AWS WAF updated the Windows operating system rule set.	September 23, 2020
Updated AWS Managed Rules for AWS WAF (p. 47)	AWS Managed Rules for AWS WAF updated the rule sets PHP application and POSIX operating system.	September 16, 2020
Updated AWS Shield console (p. 333)	AWS Shield offers a new console option, with an improved user experience. The console guidance in the documentation is for the new console.	September 1, 2020
Firewall Manager updates to common security group policies (p. 292)	AWS Firewall Manager common security group policies now support Application Load Balancers and Classic Load Balancers resource types through the console implementation. The new options are available in the common policy's **Policy scope** settings.	August 11, 2020
Updated AWS Managed Rules for AWS WAF (p. 47)	AWS Managed Rules for AWS WAF updated the core rule set.	August 7, 2020
Firewall Manager supports AWS WAF logging configuration (p. 285)	AWS Firewall Manager now supports centralized logging configuration for AWS WAF policies.	July 30, 2020
Specify IP address location in web request (p. 82)	Added the option to use IP addresses from an HTTP header that you specify, instead of using the web request origin. The alternate header is commonly `X-Forwarded-For` (XFF), but you can specify any header name. You can use this option for IP set matching, geo matching, and rate-based rule count aggregation.	July 9, 2020
Firewall Manager updates to content audit security group policies (p. 289)	AWS Firewall Manager has expanded functionality for content audit security group policies including a managed rules option, that uses managed application and protocol lists, and details for resource violations.	July 7, 2020

Firewall Manager managed lists (p. 281)	AWS Firewall Manager now supports managed application and protocol lists. Firewall Manager manages some lists and you can create and manage your own.	July 7, 2020
Add support for AWS Shield Advanced proactive engagement (p. 340)	You can configure Shield Advanced to have the Shield Response Team (SRT) contact you if the Amazon Route 53 health check associated with a protected resource becomes unhealthy during an event that's detected by Shield Advanced.	June 8, 2020
Firewall Manager supports shared VPCs in common security group policies (p. 287)	AWS Firewall Manager now supports using common security group policies in shared VPCs. You can do this in addition to using them in the VPCs owned by in-scope accounts.	May 26, 2020
Updated AWS Managed Rules for AWS WAF (p. 38)	Added documentation for each rule in the AWS Managed Rules for AWS WAF.	May 19, 2020
Updated AWS Managed Rules for AWS WAF (p. 47)	AWS Managed Rules for AWS WAF updated the Linux operating system rule group.	May 19, 2020
Add support for migrating AWS WAF Classic resources to AWS WAF (v2) (p. 11)	You can now use the console or API to export your AWS WAF Classic resources for migration to the latest version of AWS WAF.	April 27, 2020
Add support for AWS Organizations organizational units in policy scope (p. 265)	AWS Firewall Manager now supports using AWS Organizations organizational units (OUs) to specify policy scope. You can use OUs to include or exclude accounts from the scope, in addition to including or excluding specific accounts. Specifying an OU is the same as specifying all accounts in the OU and in any of its child OUs, including any child OUs and accounts that are added at a later time.	April 6, 2020
Add support for AWS WAF (v2) to AWS Firewall Manager (p. 265)	AWS Firewall Manager now supports the latest version of AWS WAF, in addition to the prior version, AWS WAF Classic.	March 31, 2020

Update to AWS Firewall Manager common security group policies	AWS Firewall Manager common security group policy now has the option to apply the policy to all elastic network interfaces in your in-scope Amazon EC2 instances. You can still choose to only apply the policy to the default elastic network interface.	March 11, 2020
Updated AWS Managed Rules for AWS WAF	AWS Managed Rules for AWS WAF added an `AWSManagedRulesAnonymousIpList` rule group.	March 6, 2020
Updated AWS Managed Rules for AWS WAF	AWS Managed Rules for AWS WAF updated the WordPress application and `AWSManagedRulesCommonRuleSet` rule groups.	March 3, 2020
Added Amazon Route 53 health check to AWS Shield Advanced protection options (p. 327)	Shield Advanced now supports the use of Amazon Route 53 health check associations, to improve the accuracy of threat detection and mitigation.	February 14, 2020
Updated AWS Managed Rules for AWS WAF	AWS Managed Rules for AWS WAF has updated the SQL Database rule group to add checking the message URI.	January 23, 2020
Firewall Manager new option for security group usage audit policy	Firewall Manager has a new option for security group usage audit policies. You can now set a minimum number of minutes a security group must remain unused before it's considered noncompliant. By default, this minutes setting is zero.	January 14, 2020
Firewall Manager new option for AWS WAF policy	Firewall Manager has a new option for AWS WAF policies. You can now choose to remove all existing web ACL associations from in-scope resources before associating the policy's new web ACLs to them.	January 14, 2020
Updated AWS Managed Rules for AWS WAF	AWS Managed Rules for AWS WAF has updated text transformations for rules in the Core Rule Set and the SQL Database rule groups.	December 20, 2019

AWS Firewall Manager integrated with AWS Security Hub	AWS Firewall Manager now creates findings for resources that are out of compliance and for attacks and sends them to AWS Security Hub.	December 18, 2019
Release of AWS WAF version 2	New version of the AWS WAF developer guide. You can manage a web ACL or rule group in JSON format. Expanded capabilities include logical rule statements, rule statement nesting, and full CIDR support for IP addresses and address ranges. Rules are no longer AWS resources, but exist only in the context of a web ACL or rule group. For existing customers, the prior version of the service is now called AWS WAF Classic. In the APIs, SDKs, and CLIs, AWS WAF Classic retains its naming schemes and this latest version of AWS WAF is referred to with an added "V2" or "v2", depending on the context. AWS WAF can't access AWS resources that were created in AWS WAF Classic. To use those resources in AWS WAF, you need to migrate them.	November 25, 2019
AWS Managed Rules rule groups for AWS WAF	Added AWS Managed Rules rule groups. These are free of charge for AWS WAF customers.	November 25, 2019
AWS Firewall Manager support for Amazon Virtual Private Cloud security groups	Added support for Amazon VPC security groups to Firewall Manager.	October 10, 2019
AWS Firewall Manager support for AWS Shield Advanced	Added support for Shield Advanced to Firewall Manager.	March 15, 2019
Tutorial: Creating hierarchical policies	Added tutorial on creating hierarchical policies in AWS Firewall Manager.	February 11, 2019
Rule-level control in rule groups	You can now exclude individual rules from AWS Marketplace rule groups, as well as your own rule groups.	December 12, 2018
AWS Shield Advanced support for AWS Global Accelerator	Shield Advanced can now protect AWS Global Accelerator.	November 26, 2018
AWS WAF support for Amazon API Gateway	AWS WAF now protects Amazon API Gateway APIs.	October 25, 2018

Expanded AWS shield advanced getting started wizard	New wizard provides opportunity to create rate-based rules and Amazon CloudWatch Events.	August 31, 2018
AWS WAF logging	Enable logging to get detailed information about traffic that is analyzed by your web ACL.	August 31, 2018
Support for query parameters in conditions	When creating a condition, you can now search the requests for specific parameters.	June 5, 2018
Shield advanced getting started wizard	Introduces a new streamlined process for subscribing to AWS Shield Advanced.	June 5, 2018
Expanded allowed CIDR ranges	When creating an IP match condition, AWS WAF now supports IPv4 address ranges: /8 and any range between /16 through /32.	June 5, 2018

Earlier updates

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	AWS Marketplace rule groups	November, 2017
Update	2016-08-24	Shield Advanced support for Elastic IP addresses	November, 2017
Update	2016-08-24	Global threat dashboard	November, 2017
Update	2016-08-24	DDoS-resistant website tutorial	October, 2017
Update	2016-08-24	Geo and regex conditions	October, 2017
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
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	April 28, 2016

Change	API Version	Description	Release Date
		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. If you are already using AWS CloudTrail, you will start seeing AWS WAF API calls in your CloudTrail log. If you haven't enabled CloudTrail for your account, you can enable it on CloudTrail from the AWS Management Console. There is no additional charge for enabling CloudTrail, but standard rates for Amazon S3 and Amazon SNS usage apply.	
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 Cross-site scripting attack rule statement (p. 74).	March 29, 2016
New Features	2015-08-24	With this release, AWS WAF adds the following features: • 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 Size constraint rule statement (p. 71). • 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 Request component (p. 75).	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*.