AWS Network Firewall

Developer Guide

AWS Network Firewall: Developer Guide

Table of Contents

What is AWS Network Firewall?

AWS Network Firewall is a stateful, managed, network firewall and intrusion detection and prevention service for your virtual private cloud (VPC) that you created in Amazon Virtual Private Cloud (Amazon VPC).

With Network Firewall, you can filter traffic at the perimeter of your VPC. This includes filtering traffic going to and coming from an internet gateway, NAT gateway, or over VPN or AWS Direct Connect. Network Firewall uses the open source intrusion prevention system (IPS), Suricata, for stateful inspection. Network Firewall supports Suricata compatible rules. For more information, see Stateful rule groups in AWS Network Firewall (p. 41).

You can use Network Firewall to monitor and protect your Amazon VPC traffic in a number of ways, including the following:

- Pass traffic through only from known AWS service domains or IP address endpoints, such as Amazon S3.
- Use custom lists of known bad domains to limit the types of domain names that your applications can access.
- Perform deep packet inspection on traffic entering or leaving your VPC.
- Use stateful protocol detection to filter protocols like HTTPS, independent of the port used.

To enable Network Firewall for your VPC, you perform steps in both Amazon VPC and in Network Firewall. For information about managing your Amazon Virtual Private Cloud VPC, see the Amazon Virtual Private Cloud User Guide. For more information about how Network Firewall works, see How AWS Network Firewall works (p. 5).

Network Firewall is supported by AWS Firewall Manager. You can use Firewall Manager to centrally configure and manage your firewalls across your accounts and applications in AWS Organizations. You can manage firewalls for multiple accounts using a single account in Firewall Manager. For more information, see AWS Firewall Manager in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

Topics

AWS Network Firewall AWS resources

Network Firewall manages the following AWS resource types:

- **Firewall** – Provides traffic filtering logic for the subnets in a VPC.

- **FirewallPolicy** – Defines rules and other settings for a firewall to use to filter incoming and outgoing traffic in a VPC.
- **RuleGroup** – Defines a set of rules to match against VPC traffic, and the actions to take when Network Firewall finds a match. Network Firewall uses stateless and stateful rule group types, each with its own Amazon Resource Name (ARN).

AWS Network Firewall concepts

AWS Network Firewall is a firewall service for Amazon Virtual Private Cloud (Amazon VPC). For information about managing your Amazon Virtual Private Cloud VPC, see the Amazon Virtual Private Cloud User Guide.

The following are the key concepts for Network Firewall:

- **Virtual private cloud (VPC)** – A virtual network dedicated to your AWS account.
- **Internet gateway** – A gateway that you attach to your VPC to enable communication between resources in your VPC and the internet.
- **Subnet** – A range of IP addresses in your VPC. Network Firewall creates firewall endpoints in subnets inside your VPC, to filter network traffic. In a VPC architecture that uses Network Firewall, the firewall endpoints sit between your protected subnets and locations outside your VPC.
- **Firewall subnet** – A subnet that you've designated for exclusive use by Network Firewall for a firewall endpoint. A firewall endpoint can't filter traffic coming into or going out of the subnet in which it resides, so don't use your firewall subnets for anything other than Network Firewall.
- **Route table** – A set of rules, called routes, that are used to determine where network traffic is directed. You modify your VPC route tables in Amazon VPC to direct traffic through your firewalls for filtering.
- **Network Firewall *firewall*** – An AWS resource that provides traffic filtering logic for the subnets in a VPC.
- **Network Firewall *firewall policy*** – An AWS resource that defines rules and other settings for a firewall to use to filter incoming and outgoing traffic in a VPC.
- **Network Firewall *rule group*** – An AWS resource that defines a set of rules to match against VPC traffic, and the actions to take when Network Firewall finds a match.
- **Stateless rules** – Criteria for inspecting a single network traffic packet, without the context of the other packets in the traffic flow, the direction of flow, or any other information that's not provided by the packet itself.
- **Stateful rules** – Criteria for inspecting network traffic packets in the context of their traffic flow.

Accessing AWS Network Firewall

You can create, access, and manage your firewall, firewall policy, and rule group resources in Network Firewall using any of the following methods:

- **AWS Management Console** – Provides a web interface for managing the service. The procedures throughout this guide explain how to use the AWS Management Console to perform tasks for Network Firewall. You can access the AWS Management Console at https://aws.amazon.com/console. To access Network Firewall using the console:

```
https://<region>.console.aws.amazon.com/network-firewall/home
```

- **AWS Command Line Interface (AWS CLI)** – Provides commands for a broad set of AWS services, including Network Firewall. The CLI is supported on Windows, macOS, and Linux. For more

information, see the AWS Command Line Interface User Guide. To access Network Firewall using the CLI endpoint:

```
aws network-firewall
```

- **AWS Network Firewall API** – Provides a RESTful API. The REST API requires you to handle connection details, such as calculating signatures, handling request retries, and handling errors. For more information, see AWS APIs and the AWS Network Firewall API Reference. To access Network Firewall, use the following REST API endpoint:

```
https://network-firewall.<region>.amazonaws.com
```

- **AWS SDKs** – Provide language-specific APIs. If you're using a programming language that AWS provides an SDK for, you can use the SDK to access AWS Network Firewall. The SDKs handle many of the connection details, such as calculating signatures, handling request retries, and handling errors. They integrate easily with your development environment, and provide easy access to Network Firewall commands. For more information, see Tools for Amazon Web Services.
- **AWS CloudFormation** – Helps you model and set up your Amazon Web Services resources so that you can spend less time managing those resources and more time focusing on your applications that run in AWS. You create a template that describes all the AWS resources that you want and AWS CloudFormation takes care of provisioning and configuring those resources for you. For more information, see Network Firewall resource type reference in the *AWS CloudFormation User Guide*.
- **AWS Tools for Windows PowerShell** – Let developers and administrators manage their AWS services and resources in the PowerShell scripting environment. For more information, see the AWS Tools for Windows PowerShell User Guide.

Regions and endpoints for AWS Network Firewall

To reduce data latency in your applications, AWS Network Firewall offers a regional endpoint to make your requests:

```
https://network-firewall.<region>.amazonaws.com
```

To view the complete list of AWS Regions where Network Firewall is available, see Service endpoints and quotas in the *AWS General Reference*.

Pricing for AWS Network Firewall

For detailed information about pricing for Network Firewall, see AWS Network Firewall pricing.

Some configurations can incur additional costs, on top of the basic costs for using Network Firewall. For example, if you use a firewall endpoint in one Availability Zone to filter traffic from another zone, you can incur cross-zone traffic charges. If you enable logging, you incur additional charges according to factors such as the logging destination that you use and the amount of traffic that you choose to log.

AWS Network Firewall quotas

AWS Network Firewall defines maximum settings and other quotas on the number of Network Firewall resources that you can use. You can request an increase for some of these quotas. For more information, see AWS Network Firewall quotas (p. 101).

AWS Network Firewall additional resources

To get a hands-on introduction to AWS Network Firewall, complete Getting started with AWS Network Firewall (p. 22).

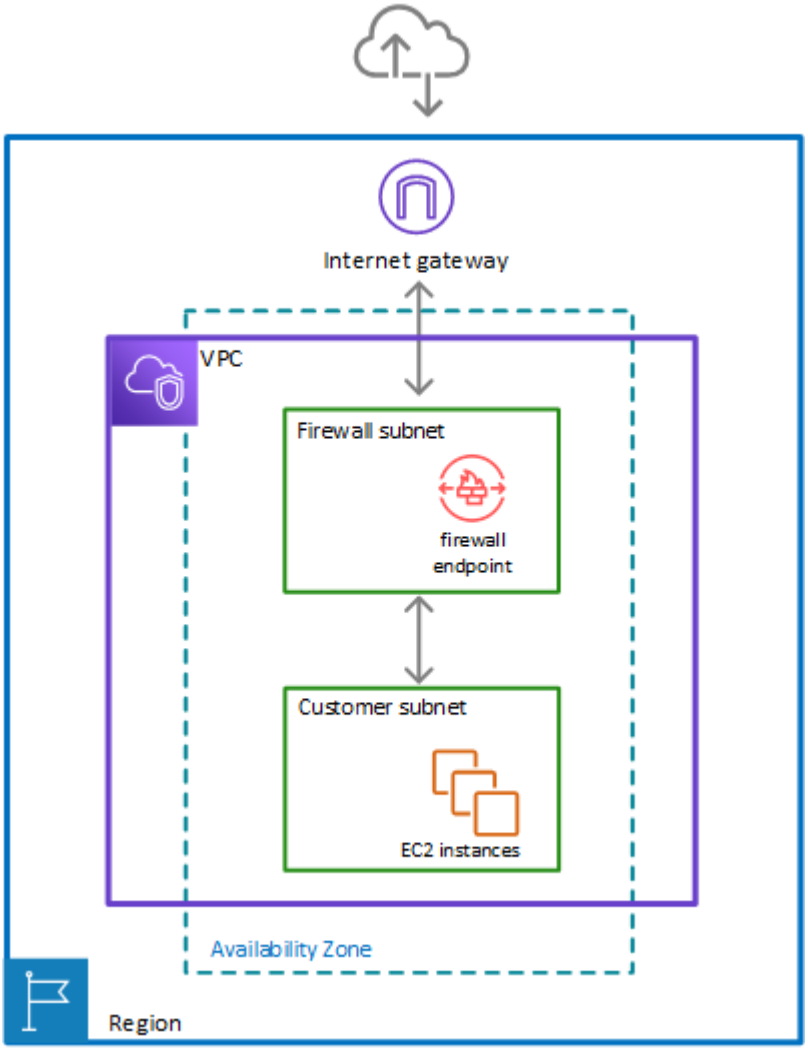
Use the following resources to get additional information and guidance for using AWS Network Firewall.

- **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 Support center** – The home page for AWS Support.
- **Contact Us** – A central contact point for inquiries concerning billing, accounts, and events.

How AWS Network Firewall works

AWS Network Firewall is a stateful, managed, network firewall and intrusion detection and prevention service for Amazon Virtual Private Cloud (Amazon VPC). You can combine Network Firewall with services and components that you use with your VPC, for example an internet gateway, a NAT gateway, a VPN, or a transit gateway. For information about managing your Amazon Virtual Private Cloud VPC, see the Amazon Virtual Private Cloud User Guide. You need a VPC to use Network Firewall.

The firewall protects the subnets within your VPC by filtering traffic going between the subnets and locations outside of your VPC. The following example figure depicts the placement of a firewall in a very simple architecture.



To enable the firewall's protection, you modify your Amazon VPC route tables to send your network traffic through the Network Firewall firewall endpoints. For information about managing route tables for your VPC, see Route tables in the *Amazon Virtual Private Cloud User Guide*.

Firewall components in AWS Network Firewall

The AWS Network Firewall firewall runs stateless and stateful traffic inspection rules engines. The engines use rules and other settings that you configure inside a firewall policy.

You install the firewall endpoints on a per-Availability Zone basis in your VPC. For each Availability Zone where you want an endpoint, you choose a subnet to host it. The firewall endpoint can protect any subnet in your VPC except for the one in which it's located.

You manage Network Firewall firewalls with the following central components.

- **Rule group** – Holds a reusable collection of criteria for inspecting traffic and for handling packets and traffic flows that match the inspection criteria. For example, you can choose to drop or pass a packet or all packets in a traffic flow based on the inspection criteria. Some rule groups fully define the behavior and some use lower-level rules that provide more detail. Rule groups are either stateless or stateful. For more information about rule groups and rules, see Rule groups in AWS Network Firewall (p. 39).
- **Firewall policy** – Defines a reusable set of stateless and stateful rule groups, along with some policy-level behavior settings. The firewall policy provides the network traffic filtering behavior for a firewall. You can use a single firewall policy in multiple firewalls. For more information about firewall policies, see Firewall policies in AWS Network Firewall (p. 34).
- **Firewall** – Connects the inspection rules in the firewall policy to the VPC that the rules protect. Each firewall requires one firewall policy. The firewall additionally defines settings like how to log information about your network traffic and the firewall's stateful traffic filtering. For more information about firewalls, see Firewalls in AWS Network Firewall (p. 30).

High-level steps for implementing a firewall

To install and use an AWS Network Firewall firewall in your Amazon Virtual Private Cloud VPC, you configure the firewall components and your VPC's subnets and route tables in the following high-level steps.

- **Configure the VPC subnets for your firewall endpoints** – In your VPC, in each Availability Zone where you want a firewall endpoint, create a subnet specifically for use by Network Firewall. A firewall endpoint can't protect applications that run in the same subnet, so reserve these subnets for exclusive use by the firewall. The subnets that you use for your firewall endpoints must belong to a single AWS Region and must be in different Availability Zones within the Region. Network Firewall is available in the Regions listed at AWS service endpoints.

  For information about managing subnets in your VPC, see VPCs and subnets in the *Amazon Virtual Private Cloud User Guide*.
- **Create the firewall** – Create a Network Firewall firewall and provide it with the specifications for each of your firewall subnets. Network Firewall creates a firewall endpoint in each subnet that you specify, available to monitor and protect the resources for the subnets whose traffic you send through it.
- **Configure the firewall policy** – Define the firewall policy for your firewall by specifying its rule groups and other behavior that you want the firewall to provide.
- **Modify your VPC route tables to include the firewall** – Using Amazon VPC ingress routing enhancements, change your routing tables to route traffic through the Network Firewall firewall. These changes must insert the firewall 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.

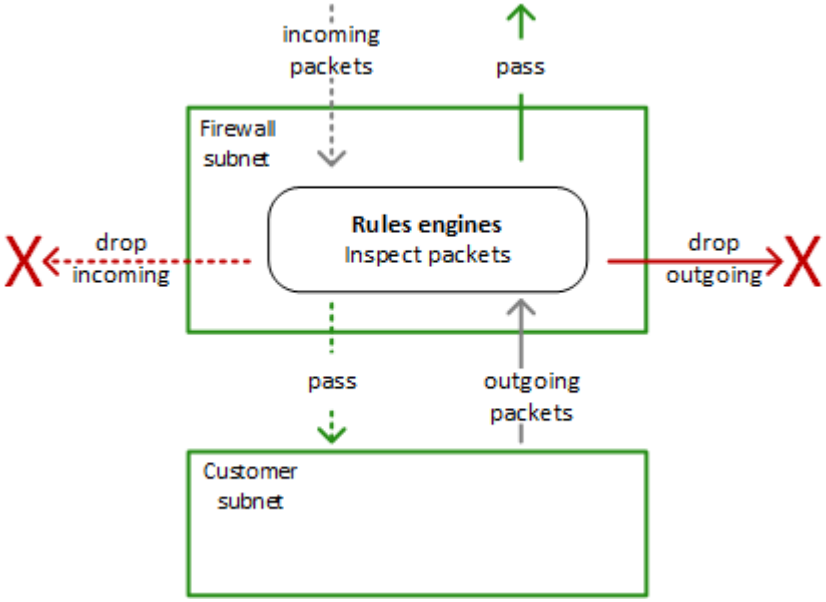
  For information about managing route tables for your VPC, see Route tables in the *Amazon Virtual Private Cloud User Guide*.

Firewall behavior in AWS Network Firewall

AWS Network Firewall provides virtual firewalls dedicated to protecting your VPC from attacks. You define and create a firewall, then use it to monitor and protect your subnets. The firewall monitors incoming and outgoing traffic and allows it to pass or drops it, according to your specifications. The firewall only allows packets to pass that pass inspection.

Network Firewall monitors and controls traffic to and from your protected subnets

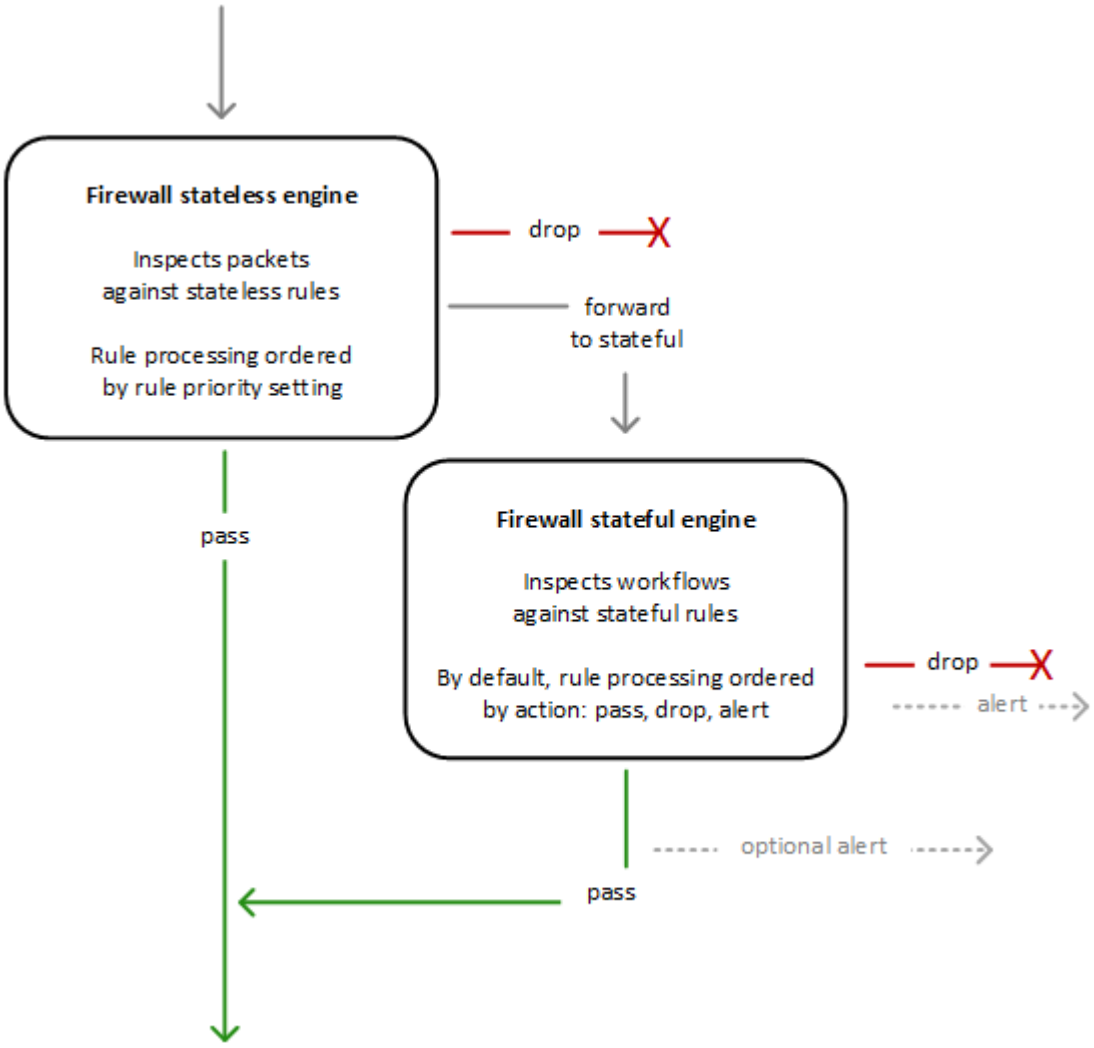
The following figure shows the basic interaction of your firewall with traffic coming into your customer subnet and with traffic going out from your customer subnet.



Network Firewall stateless and stateful rules engines

AWS Network Firewall uses two rules engines to inspect packets. The engines inspect packets according to the rules that you provide in your firewall policy.

The following figure shows the processing flow for packets coming through the firewall. First the stateless engine inspects the packet against the configured stateless rules. Depending on the packet settings, the stateless inspection criteria, and the firewall policy settings, the stateless engine might drop a packet, pass it through to its destination, or forward it to the stateful rules engine. The stateful engine inspects packets in the context of their traffic flow, using the configured stateful rules. The stateful engine either drops packets or passes them to their destination. Stateful engine activities send flow and alert logs to the firewall's logs, if logging is configured. The stateful engine sends alerts for dropped packets and can optionally send them for passed packets.



The stateless and stateful rules inspection engines operate in different ways:

- **Stateless rules engine** – Inspects each packet in isolation, without regard to factors such as the direction of traffic, or whether the packet is part of an existing, approved connection. This engine prioritizes the speed of evaluation. It takes rules with standard 5-tuple connection criteria. The engine processes your rules in the order that you prioritize them and stops processing when it finds a match.

  Network Firewall stateless rules are similar in behavior and use to Amazon VPC network access control lists (ACLs).

- **Stateful rules engine** – Inspects packets in the context of their traffic flow, allows you to use more complex rules, and allows you to log network traffic and to log Network Firewall firewall alerts on traffic. Stateful rules consider traffic direction. The stateful rules engine might delay packet delivery in order to group packets for inspection. By default, the stateful rules engine processes your rules in the order of their action setting, with pass rules processed first, then drop, then alert. The engine stops processing when it finds a match.

  The stateful engine takes rules that are compatible with Suricata, an open source intrusion prevention system (IPS). Suricata provides a standard rule-based language for stateful network traffic inspection. For more information about Suricata, see Stateful rule groups in AWS Network Firewall (p. 41) and the Suricata website.

Network Firewall stateful rules are similar in behavior and use to Amazon VPC security groups. By default, the stateful rules engine allows traffic to pass, while the security groups default is to deny traffic.

Whether you use only one of these engines or a combination depends on your specific use case.

How AWS Network Firewall filters network traffic

When AWS Network Firewall inspects a packet, it evaluates the packet against the rules in the policy's stateless rule groups first, using the stateless rules engine. Then, depending on that inspection and on other settings in the policy, it might evaluate the packets against the rules in the policy's stateful rule groups, using the stateful rules engine.

1. Stateless rules engine

Network Firewall evaluates each packet against the firewall policy's stateless rules until it finds a match or exhausts all of the stateless rules. Network Firewall evaluates the rule groups in the order that they are prioritized in the policy, starting from the lowest setting. Within each rule group, Network Firewall evaluates the rules in the order that they are prioritized in the rule group, starting from the lowest setting. When you create a stateless rule group, you set the priority of the rules in the rule group. When you create a firewall policy, you set the priority of the stateless rule groups in the policy. For more information, see Stateless rule groups in AWS Network Firewall (p. 40) and Firewall policies in AWS Network Firewall (p. 34).

When Network Firewall finds a match, it handles the packet according to the matching rule's configuration. You configure a stateless rule to pass the packet through, drop it, or forward it to your stateful rules. Additionally, you can configure a stateless rule to perform a custom action, for example you can publish metrics for the packet to Amazon CloudWatch. For more information, see Rule actions in AWS Network Firewall (p. 55).

2. Default stateless rule actions

If a packet doesn't match any stateless rule, Network Firewall performs the firewall policy's default stateless rule action for full packet or UDP packet fragment, depending on the packet type. Network Firewall only applies the fragment action setting to UDP packet fragments, and silently drops packet fragments for other protocols. The options for these actions settings are the same as for stateless rules. For more information, see Stateless default actions in your firewall policy (p. 35).

3. Stateful rules engine

When Network Firewall forwards a packet to the stateful engine for inspection, it inspects each packet against the stateful rule groups, in the context of the packet's traffic flow. You can configure a stateful rule to pass the packet through, with or without an alert, or drop it and send an alert. Alerts require logging to be configured for the firewall.

The Suricata stateful rules engine controls how the stateful rules in your firewall policy are processed. The engine evaluates the packet's traffic flow against the conditions in the policy's stateful rules until it finds a match or exhausts all of the rules. When the engine finds a match, it handles the packet according to the rule's configuration. By default, the Suricata stateful rules engine orders rule processing according to the rule action setting, processing first the rules with pass action, then drop, then alert. For more information, see Rule actions in AWS Network Firewall (p. 55) and the Suricata Action-order documentation.

Depending on the Suricata compatible rules that you provide, the stateful engine might perform deep packet inspection of your traffic. Deep packet inspection works on the payload data within your packets, rather than on the header information.

For more information about stateful rules, see Rule groups in AWS Network Firewall (p. 39).

Route table configurations for AWS Network Firewall

To include the firewall in your Amazon Virtual Private Cloud VPC, you modify the VPC route tables so that the traffic that you want the firewall to filter passes through the firewall endpoints. Exactly how you do this depends on your architecture and the traffic that you want to filter. For example, to filter all traffic between an internet gateway and your customer subnets, you redirect incoming traffic from the internet gateway and outgoing traffic from the customer subnets through the firewall endpoint.

For information about managing route tables for your VPC, see Route tables in the *Amazon Virtual Private Cloud User Guide*.

For descriptions of common architectures for AWS Network Firewall, with example route table configurations, see AWS Network Firewall example architectures with routing (p. 10).

AWS Network Firewall example architectures with routing

This section provides a high-level view of simple architectures that you can configure with AWS Network Firewall and shows example route table configurations for each. For additional information and examples, see Deployment models for AWS Network Firewall.

> **Note**
> For information about managing route tables for your VPC, see Route tables in the *Amazon Virtual Private Cloud User Guide*.

Unsupported architectures

The following lists architectures and traffic types that Network Firewall doesn't support:

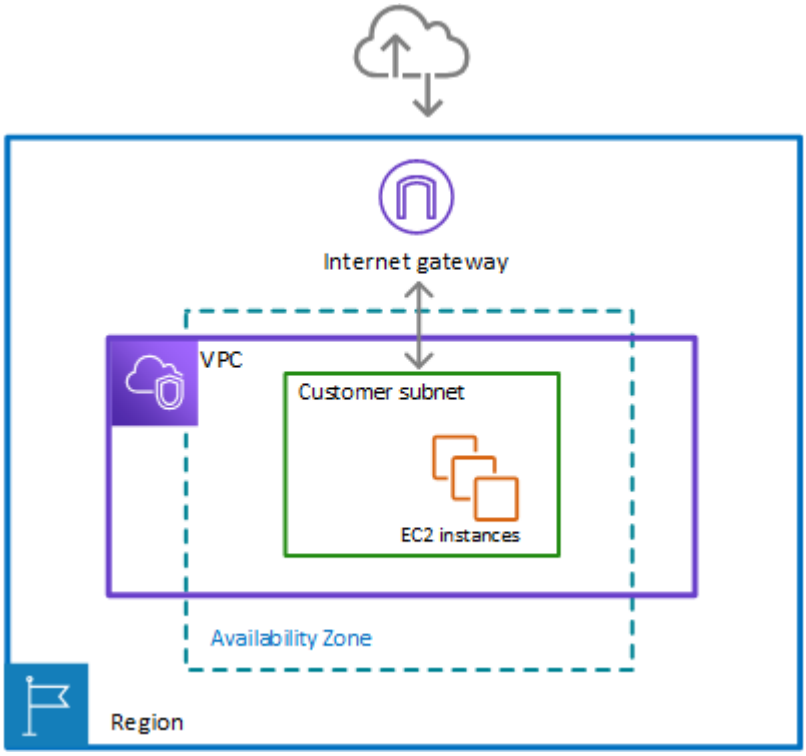
- VPC peering.
- Virtual private gateways.
- Inspection of AWS Global Accelerator traffic.
- Inspection of AmazonProvidedDNS traffic for Amazon EC2.

Simple single zone architecture with an internet gateway

This topic provides a high-level view of a simple VPC configuration using an internet gateway and AWS Network Firewall. It describes the basic route table modifications that are required to use the firewall.

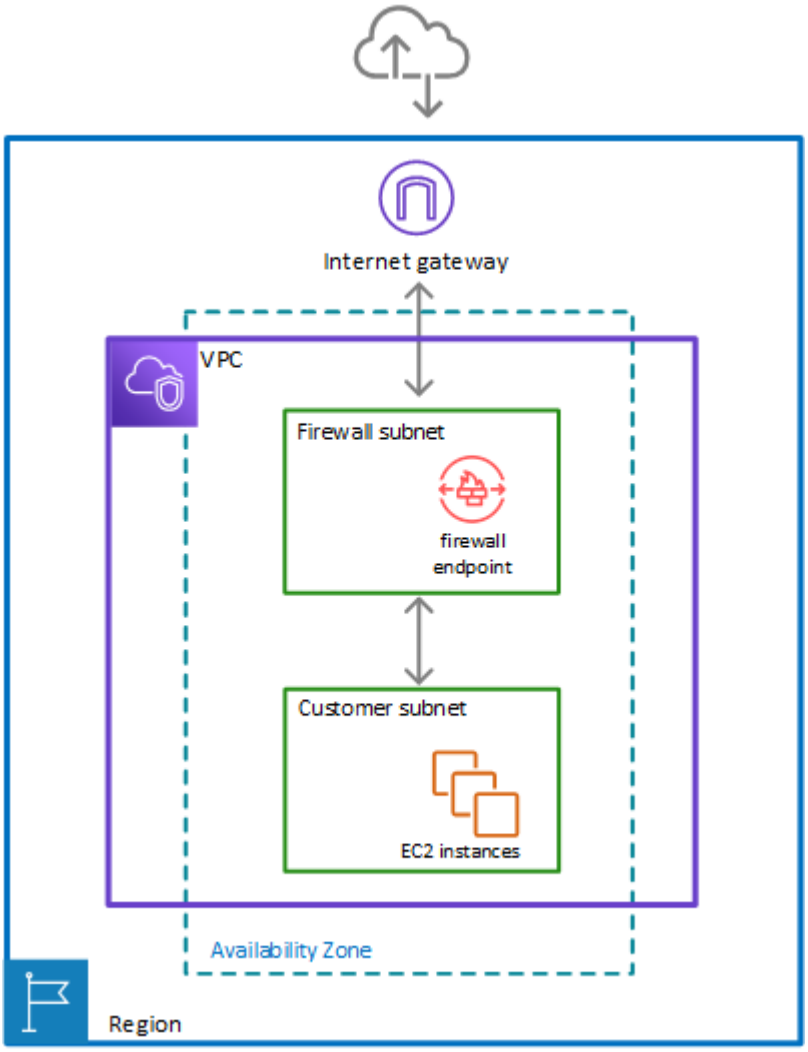
Single zone architecture with internet gateway and no firewall

The following figure depicts a simple VPC configuration with a single customer subnet, and no firewall. The VPC has an internet gateway for internet access. All incoming and outgoing traffic routes through the internet gateway to the subnet.



Single zone architecture with internet gateway and the Network Firewall firewall

The following figure depicts a simple VPC configuration with the firewall and the subnet association in place. The VPC has an internet gateway for internet access. All incoming and outgoing traffic for the VPC routes through the firewall.

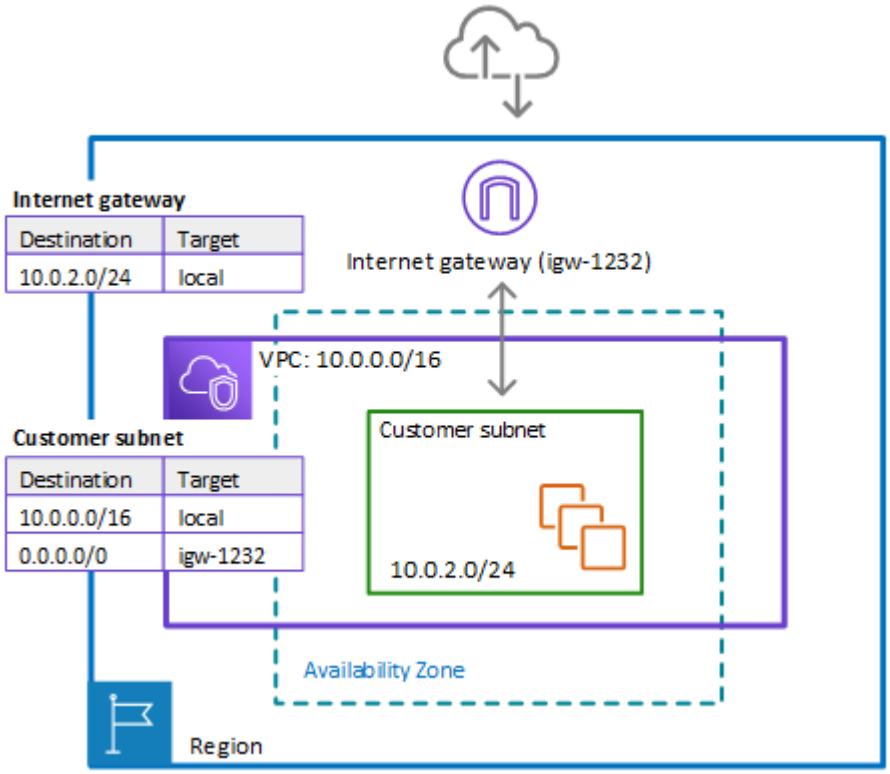


To include the firewall in your Amazon Virtual Private Cloud VPC, you need to modify the VPC route tables so that traffic between the customer subnets and the internet passes through the firewall, for both incoming and outgoing traffic.

> **Note**
> For information about managing route tables for your VPC, see Route tables in the *Amazon Virtual Private Cloud User Guide*.

Example route tables in the single zone architecture with no firewall

The following figure depicts the route tables that provide the correct flow of traffic for a single Availability Zone without a firewall:

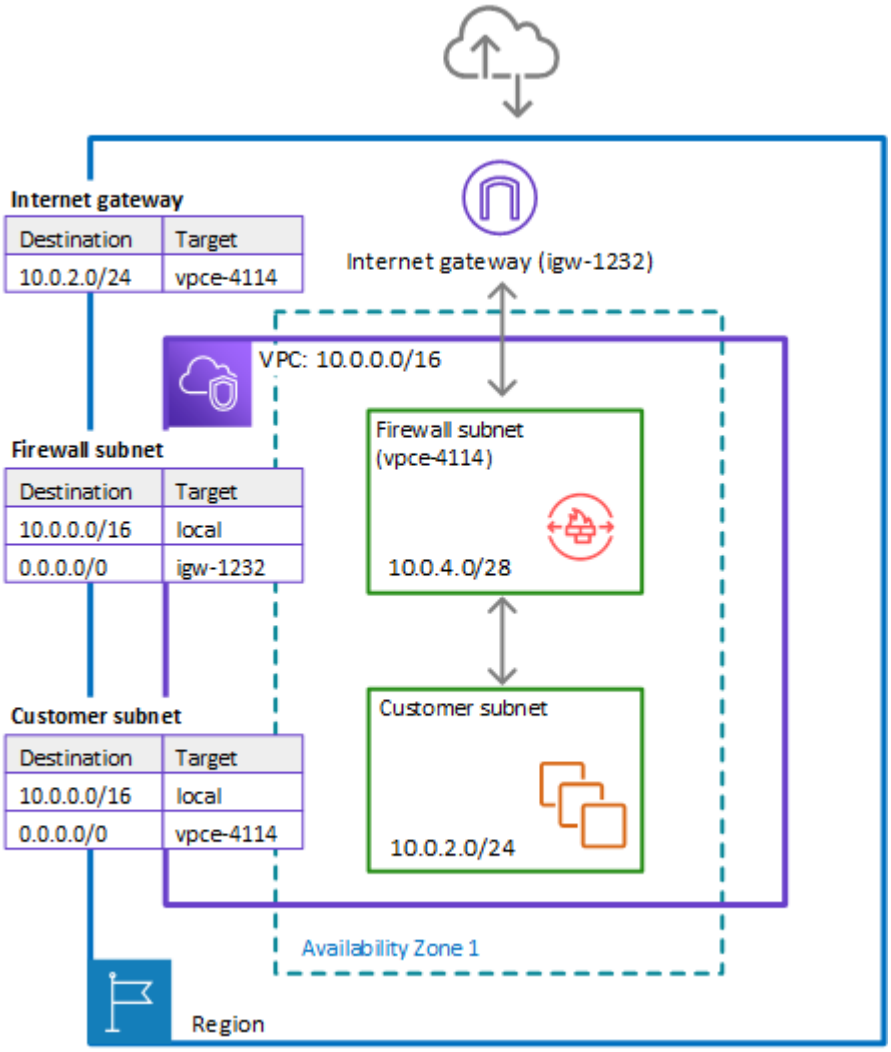


In the preceding figure, the route tables enforce the following traffic flows:

- **Internet gateway route table** – Routes traffic that's destined for the customer subnet (range `10.0.2.0/24`) to `local`. The customer subnet shows the private IP address range behind the publicly assigned address. The subnet has public addresses assigned, which are either auto-generated or assigned via Elastic IP address. Within a VPC, only private IP addresses are used for communication.
- **Customer subnet route table** – Routes traffic that's destined for anywhere inside the VPC (`10.0.0.0/16`) to the local address. Routes traffic that's destined for anywhere else (`0.0.0.0/0`) to the internet gateway (`igw-1232`).

Example route tables in the single zone architecture with the firewall

The following figure depicts the same installation with the Network Firewall firewall added and the route tables changed to include the firewall. The route tables direct traffic between the customer subnet and the internet gateway through the firewall endpoint:



In the preceding figure, the route tables enforce the following traffic flows:

- **Internet gateway route table** – Routes traffic that's destined for the customer subnet (range `10.0.2.0/24`) to the firewall subnet (named `vpce-4114` in the figure). The customer subnet shows the private IP address range behind the publicly assigned address. The subnet has public addresses assigned, which are either auto-generated or assigned via Elastic IP address. Within a VPC, only private IP addresses are used for communication.
- **Firewall subnet route table** – Routes traffic that's destined for anywhere inside the VPC (`10.0.0.0/16`) to the local address. Routes traffic that's destined for anywhere else (`0.0.0.0/0`) to the internet gateway (`igw-1232`).
- **Customer subnet route table** – Routes traffic that's destined for anywhere inside the VPC (`10.0.0.0/16`) to the local address. Routes traffic that's destined for anywhere else (`0.0.0.0/0`) to the firewall subnet (`vpce-4114`).

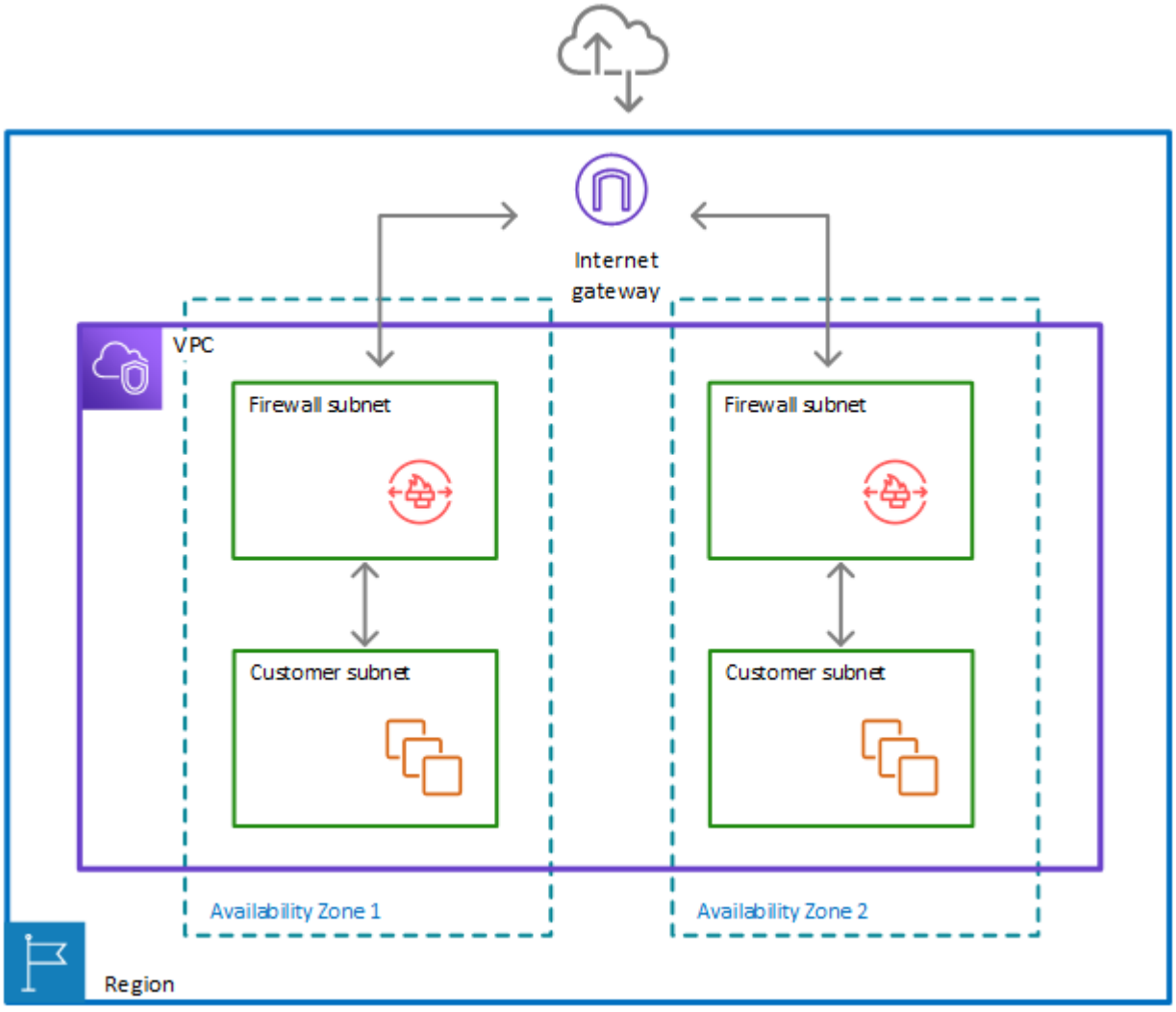
  Before the firewall inclusion, the customer subnet route table routed the `0.0.0.0/0` traffic to `igw-1232`.

Multi zone architecture with an internet gateway

This topic provides a high-level view of a simple two zone VPC configuration using an internet gateway and AWS Network Firewall. It describes the basic route table modifications that are required to use the Network Firewall firewall.

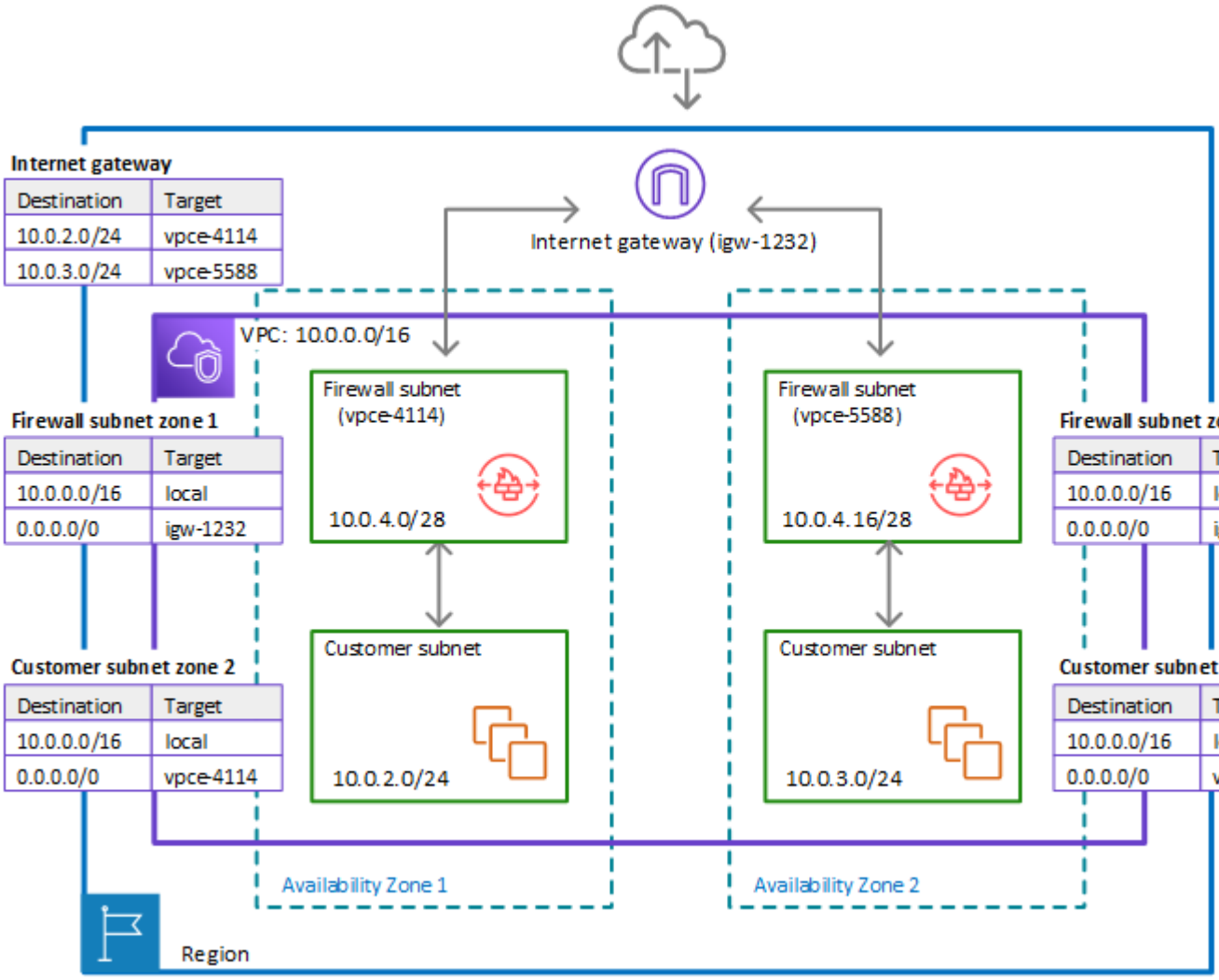
Two zone architecture with internet gateway and the Network Firewall firewall

The following figure depicts a Network Firewall configuration for a VPC that spans multiple Availability Zones. In this case, each Availability Zone that the VPC spans has a firewall subnet and a customer subnet. The VPC has an internet gateway for internet access. All incoming traffic for the VPC routes to the firewall in the same Availability Zone as the destination customer subnet. All outgoing traffic routes through the firewalls.



Route tables in the two zone architecture with the firewall

The following figure depicts a VPC configuration with two Availability Zones. Each zone has its own Network Firewall firewall, which provides monitoring and protection for the subnets in the zone. You can expand this configuration to any number of zones in your VPC.



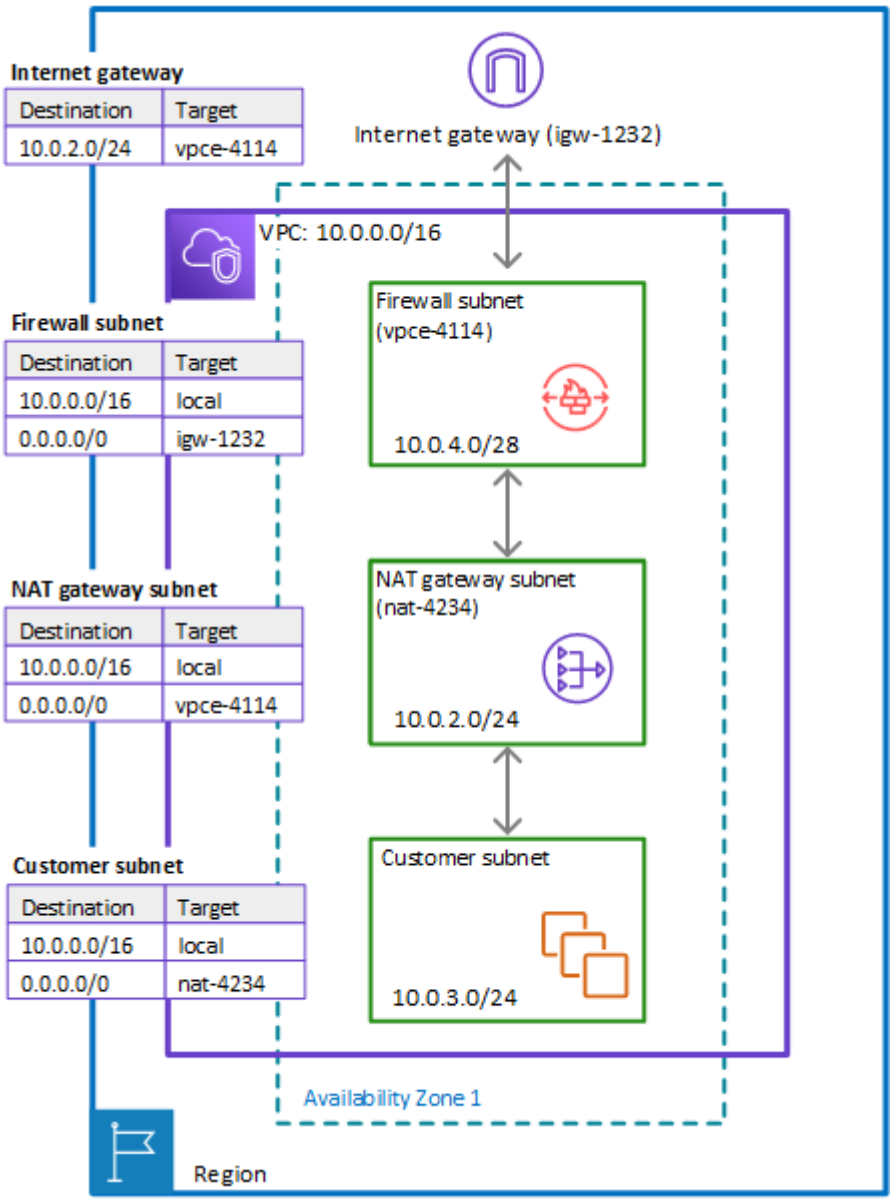
In the preceding figure, the route tables enforce similar traffic flows to the single Availability Zone model, with the primary difference being the splitting of incoming traffic by the internet gateway, to accommodate the two different customer subnets:

- **Internet gateway route table** – Routes traffic that's destined for each customer subnet (range `10.0.2.0/24` or `10.0.3.0/24`) to the firewall subnet in the same Availability Zone (`vpce-4114` or `vpce-5588`, respectively).
- **Firewall subnet route tables** – Route traffic that's destined for anywhere inside the VPC (`10.0.0.0/16`) to the local address. Route traffic that's destined for anywhere else (`0.0.0.0/0`) to the internet gateway (`igw-1232`). These are identical to the route table for the firewall subnet in the single Availability Zone.
- **Customer subnet route tables** – Route traffic that's destined for anywhere inside the VPC (`10.0.0.0/16`) to the local address. Route traffic that's destined for anywhere else (`0.0.0.0/0`) to the firewall subnet in the same Availability Zone (`vpce-4114` for zone AZ1 and `vpce-5588` for zone AZ2).

Architecture with an internet gateway and a NAT gateway

You can add a network address translation (NAT) gateway to your AWS Network Firewall architecture, for the areas of your VPC where you need NAT capabilities. AWS provides NAT gateways decoupled from your other cloud services, so you can use it in your architecture only where you need it. This can help you reduce load and load costs. For information about NAT gateways, see NAT gateways in the *Amazon Virtual Private Cloud User Guide*.

The following figure depicts a VPC configuration for Network Firewall with an internet gateway and a NAT gateway.



Setting up AWS Network Firewall

This topic describes preliminary steps, such as getting an AWS account, to prepare you to use Network Firewall. You aren't charged to set up your account or for the other preliminary items. You are charged only for AWS services that you use.

> **Note**
> Network Firewall is a network traffic firewall for your Amazon Virtual Private Cloud VPCs. If you're already working with VPCs, the setup described here shouldn't be necessary.

After you complete these steps, see Getting started with Network Firewall (p. 22) to continue getting started with Network Firewall.

Before you use Network Firewall for the first time, check that you've completed the following tasks:

- Creating an IAM user (p. 18)
- Signing in as an IAM user (p. 19)
- Creating IAM user access keys (p. 20)
- Setting up tool access (p. 20)

Get an AWS account and your root user credentials

To access AWS, you must sign up for an AWS account.

To sign up for an AWS account

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.

AWS sends you a confirmation email after the sign-up process is complete. At any time, you can view your current account activity and manage your account by going to https://aws.amazon.com/ and choosing **My Account**.

Creating an IAM user

If your account already includes an IAM user with full AWS administrative permissions, you can skip this section.

When you first create an Amazon Web Services (AWS) account, you begin with a single sign-in identity. That identity has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user*. When you sign in, enter the email address and password that you used to create the account.

> **Important**
> 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. To view the tasks that require you to sign in as the root user, see Tasks that require root user credentials.

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.

Signing in as an IAM user

Sign in to the IAM console by choosing **IAM user** and entering your AWS account ID or account alias. On the next page, enter your IAM user name and your password.

Note
For your convenience, the AWS sign-in page uses a browser cookie to remember your IAM user name and account information. If you previously signed in as a different user, choose the sign-in

link beneath the button to return to the main sign-in page. From there, you can enter your AWS account ID or account alias to be redirected to the IAM user sign-in page for your account.

Creating IAM user access keys

Access keys consist of an access key ID and secret access key, which are used to sign programmatic requests that you make to AWS. If you don't have access keys, you can create them from the AWS Management Console. As a best practice, do not use the AWS account root user access keys for any task where it's not required. Instead, create a new administrator IAM user with access keys for yourself.

The only time that you can view or download the secret access key is when you create the keys. You cannot recover them later. However, you can create new access keys at any time. You must also have permissions to perform the required IAM actions. For more information, see Permissions required to access IAM resources in the *IAM User Guide*.

To create access keys for an IAM user

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose **Users**.
3. Choose the name of the user whose access keys you want to create, and then choose the **Security credentials** tab.
4. In the **Access keys** section, choose **Create access key**.
5. To view the new access key pair, choose **Show**. You will not have access to the secret access key again after this dialog box closes. Your credentials will look something like this:

   - Access key ID: AKIAIOSFODNN7EXAMPLE
   - Secret access key: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
6. To download the key pair, choose **Download .csv file**. Store the keys in a secure location. You will not have access to the secret access key again after this dialog box closes.

   Keep the keys confidential in order to protect your AWS account and never email them. Do not share them outside your organization, even if an inquiry appears to come from AWS or Amazon.com. No one who legitimately represents Amazon will ever ask you for your secret key.
7. After you download the `.csv` file, choose **Close**. When you create an access key, the key pair is active by default, and you can use the pair right away.

Related topics

- What is IAM? in the *IAM User Guide*
- AWS security credentials in *AWS General Reference*

Setting up tool access

The AWS Management Console includes a console for Network Firewall, but if you want to access Network Firewall programmatically or through the command line, the following documentation and tools will help you:

- If you want to call the Network Firewall API without handling 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 Network Firewall 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 Network Firewall API Reference documents the operations that Network Firewall supports.
- The AWS Command Line Interface (AWS CLI) supports Network Firewall. 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 Network Firewall. For more information, see AWS Tools for PowerShell Cmdlet Reference.

Getting started with AWS Network Firewall

AWS Network Firewall provides network traffic filtering protection for your Amazon Virtual Private Cloud VPCs. This tutorial provides steps for getting started with Network Firewall using the AWS Management Console. You can also use Network Firewall API operations to create and manage your firewalls. For more information about working with Network Firewall API operations, see the AWS Network Firewall API Reference.

Topics

Before you begin

This tutorial walks you through configuring and implementing an AWS Network Firewall firewall for a VPC with a basic internet gateway architecture, like the one depicted at Simple single zone architecture with an internet gateway (p. 10).

To follow this tutorial, you'll need a test VPC where you want to implement a network firewall. Additionally, you must know how to manage the subnets and route tables in your VPC.

- For information about managing subnets in your VPC, see VPCs and subnets in the *Amazon Virtual Private Cloud User Guide*.
- For information about managing route tables for your VPC, see Route tables in the *Amazon Virtual Private Cloud User Guide*.

The test VPC that you use for this tutorial must have the following configuration in one Region:

- An internet gateway.
- A customer subnet.
- Routing configured to send inbound traffic from the internet gateway to the subnet and to send the subnet's outbound traffic to the internet gateway.
- A second subnet to use as the firewall subnet. This subnet must not be used for other purposes and must have at least one available IP address. You'll select the Availability Zone and subnet ID when you create the firewall.

If you have a different architecture that you'd like to add a firewall to, you can adjust the guidance in this tutorial accordingly. Network Firewall doesn't support some VPC architectures. For information, see AWS Network Firewall example architectures with routing (p. 10).

Step 1: Create rule groups

Rule groups are reusable collections of network filtering rules that you use to configure firewall behavior. In this step, you create a stateless rule group and a stateful rule group. For information about rule groups, see Rule groups (p. 39).

To create a stateless rule group

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Network Firewall rule groups**.
3. Choose **Create rule group**.
4. In the **Create rule group** page, for the **Rule group type**, choose **Stateless rule group**.
5. Enter the name that you want for the rule group. You'll use the name to identify the rule group when you add it to your firewall policy later in the tutorial. You can't change the name of a rule group after you create it.
6. For **Capacity**, enter **10**.
7. Enter the following rule specifications to create a stateless rule that blocks all packets coming from the source IP address CIDR range `192.0.2.0/24`:
   a. Set the priority to **10**.
   b. Leave the protocol setting at **All**.
   c. For the source address, specify **192.0.2.0/24**.
   d. Leave the source port at **Any**.
   e. Set the destination address to **Any**.
   f. For the action, choose **Drop**.
   g. Choose **Add rule**. Your rule is added to the **Rules** list.
8. Review the settings for the rule group, then choose **Create rule group**.

Your new rule group is added to the list in the **Rule groups** page.

To create a stateful rule group

1. From the **Rule groups** page, choose **Create rule group**.
2. In the **Create rule group** page, for the **Rule group type**, choose **Stateful rule group**.
3. Enter a name for the stateful rule group.
4. For **Capacity**, enter **10**.
5. Choose the stateful rule group configuration option **Import Suricata compatible rules**. The entry form for Suricata compatible IPS rules appears. Copy and paste the following Suricata rule into the text box. This rule drops TLS traffic for a specific target domain:

```
drop tls $HOME_NET any -> $EXTERNAL_NET any (tls.sni; content:"evil.com"; startswith;
 nocase; endswith; msg:"matching TLS denylisted FQDNs"; priority:1; flow:to_server,
 established; sid:1; rev:1; gid:255;)
```

6. Choose **Add rule**. Your rule is added to the **Rules** list for the rule group.
7. Review the settings for the rule group, then choose **Create rule group**.

Your stateless rule group and your stateful rule group are listed in the **Rule groups** page. You can now use these rule groups in your firewall policies.

Step 2: Create a firewall policy

Firewall policies use rule groups and other settings to define the traffic filtering behavior for a firewall. In this procedure, you'll create a policy using the rule groups that you created in the previous step. For information about firewall policies, see Firewall policies in AWS Network Firewall (p. 34).

To configure a firewall policy

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Firewall policies**.
3. In the **Firewall policies** page, choose **Create firewall policy**.
4. Enter the name that you want to use for the firewall policy. You'll use the name to identify the policy when you associate it with your firewall later in the tutorial. You can't change the name of a firewall policy after you create it.
5. Choose **Next** to go to the firewall policy's **Add rule groups** page.
6. In the **Stateless rule groups** section, choose **Add rule groups**, then select the check box for the stateless rule group that you created in the prior procedure. Choose **Add rule groups**. At the bottom of the page, the firewall policy's capacity counter shows the capacity consumed by adding this rule group next to the maximum capacity allowed for a firewall policy.
7. Your stateless rule group blocks some incoming traffic. In the stateless default actions, you choose what to do with the rest of the traffic. For this tutorial, we'll forward it to the stateful engine. Use the same default action for packets and packet fragments. Network Firewall only manages UDP packet fragments and silently drops packet fragments for other protocols. Set the action to **Forward to stateful rules**.
8. In the **Stateful rule groups** section, choose **Add rule groups**, then select the check box for the stateful rule group that you created in the prior procedure. Choose **Add rule groups**.
9. Choose **Next** then **Next** again to proceed through the tagging option and to the **Review and create** page. From this page, you can choose **Edit** for any area to return to the corresponding page in the firewall policy creation wizard.
10. Choose **Create firewall policy**.

Your new firewall policy is added to the list in the **Firewall policies** page. You can now use your firewall policy in your firewalls.

Step 3: Create a firewall

Firewalls associate the traffic filtering behavior of a firewall policy with the VPC where you want to filter traffic. In this procedure, you'll create a firewall using the firewall policy that you created in the previous step. For information about firewalls, see Firewalls in AWS Network Firewall (p. 30).

To create a firewall

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Firewalls**.
3. Choose **Create firewall**.
4. For **Name**, enter the name that you want to use to identify this firewall. You can't change the name of a firewall after you create it.

5. For **VPC**, select your VPC from the dropdown.
6. For **Availability Zone** and **Subnet**, select the zone and firewall subnet that you identified in Before you begin (p. 22).
7. For **Associated firewall policy**, choose **Associate an existing firewall policy**, then select the firewall policy that you created in the prior procedure.
8. Choose **Create firewall**.

Your new firewall is listed in the **Firewalls** page. You've configured the firewall's behavior with the firewall policy and rule groups, and your firewall has an endpoint that's running in your VPC, ready to filter network traffic.

The next step is to route the VPC's network traffic through the firewall endpoint. You'll insert it into the traffic flow between the internet gateway and your customer subnet.

Step 4: Update your Amazon VPC route tables

After you create your firewall, you insert its firewall endpoint into your Amazon Virtual Private Cloud network traffic flow, in between your internet gateway and your customer subnet. You create routing for the firewall endpoint so that it forwards traffic between the internet gateway and your subnet. Then, you update the route tables for your internet gateway and your subnet, to send traffic to the firewall endpoint instead of to each other.

This procedure covers the high-level steps for route table management. For information about managing route tables for your VPC, see Route tables in the *Amazon Virtual Private Cloud User Guide*.

To modify your route tables to insert a firewall endpoint between your internet gateway and your subnet

1. Review your routing for the internet gateway and for your customer subnet, to determine the components used to route traffic between the two.

   Record the current settings. You'll use them to reverse your changes at the end of the tutorial.

   - The internet gateway's route table typically has an entry with a destination set to your customer subnet's CIDR block and a target of `local`.
   - The subnet's route table typically has an entry with a destination set to `0.0.0.0/0` and a target set to the internet gateway ID.
2. Create a route table configuration for the firewall endpoint with the following two routes:

   - An entry that matches the internet gateway's route specification for traffic going to the customer subnet's CIDR block.
   - An entry that matches the subnet's route specification for traffic going to the internet gateway.

   The firewall endpoint is now ready to filter and forward traffic between the internet gateway and the customer subnet. The endpoint only forwards traffic to its intended destination if it passes the inspection criteria that you defined in the rule groups and firewall policy.
3. Update the internet gateway's routing to modify the entry with a destination set to your customer subnet's CIDR block. Change the target to the firewall endpoint ID.
4. Update the customer subnet routing to modify the entry with a destination set to the internet gateway ID. Change the target to the firewall endpoint ID.

The firewall endpoint is now filtering all traffic between your internet gateway and customer subnet.

Step 5: Remove the firewall and clean up your resources

You've now successfully completed the tutorial. To remove the firewall endpoint from your VPC and prevent your account from accruing AWS Network Firewall charges for the tutorial resources, revert your route table changes and clean up the Network Firewall resources that you created.

To modify your route tables to remove the firewall

1. Return the internet gateway and subnet route tables to the configurations they had at the start of the prior procedure. This stops traffic from routing to the firewall endpoint.
2. Remove the route table configuration for the firewall endpoint.

To remove the Network Firewall resources

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Firewalls**.
3. In the **Firewalls** page, select the firewall that you created for the tutorial.
4. Choose **Delete**, and then confirm your request.
5. In the navigation pane, under **Network Firewall**, choose **Firewall policies**.
6. In the **Firewall policies** page, select the firewall policy that you created for the tutorial.
7. Choose **Delete**, and confirm your request.
8. In the navigation pane, under **Network Firewall**, choose **Network Firewall rule groups**.
9. In the **Rule group** page, select the name of the rule groups that you created for the tutorial, and then choose **Delete**.

You've successfully removed the firewall from your VPC traffic flow and removed all of the Network Firewall resources that you created for this tutorial.

Configuring your VPC and other components for AWS Network Firewall

This section describes the changes that you must make in your VPC configuration and other components to use AWS Network Firewall. For information about managing your Amazon Virtual Private Cloud VPC, see the Amazon Virtual Private Cloud User Guide.

For examples of architectures that are supported by Network Firewall, see Architecture and routing examples (p. 10).

Unsupported architectures

The following lists architectures and traffic types that Network Firewall doesn't support:

- VPC peering.
- Virtual private gateways.
- Inspection of AWS Global Accelerator traffic.
- Inspection of AmazonProvidedDNS traffic for Amazon EC2.

Topics

VPC subnet configuration for AWS Network Firewall

When you associate a firewall to your VPC, you must provide a subnet for each Availability Zone where you want to place a firewall endpoint to filter traffic. A common configuration is to have a firewall endpoint in each zone where you have customer subnets that you want to protect, but you can also have a firewall endpoint filter traffic from multiple zones. When you create the firewall, Network Firewall adds a firewall endpoint to each of the designated subnets. Each firewall endpoint uses the firewall's associated firewall policy configuration to filter traffic that you route through it.

To prepare your VPC for your Network Firewall firewall, in each Availability Zone where you want a firewall endpoint, create a subnet for the endpoint. Each subnet must have at least one IP address available and a non-zero size.

> **Note**
> Reserve these firewall subnets for the exclusive use of Network Firewall. A firewall endpoint can't filter traffic coming into or going out of the subnet in which it resides, so don't place other applications in the firewall endpoint subnets.

For information about managing subnets in your VPC, see VPCs and subnets in the *Amazon Virtual Private Cloud User Guide*.

When you create your Network Firewall firewall, you must provide at least one zone and subnet for the firewall configuration. You can add and remove subnets after you create a firewall.

VPC route table configuration for AWS Network Firewall

After you create your firewall, you reroute your VPC network traffic through the firewall endpoints so they can start filtering traffic. Perform the following steps:

1. Review the route table configurations in your VPC Availability Zones for the subnets that you want to protect and for any location that sends traffic to the subnets or receives traffic from them.
2. Determine which traffic you want the firewall to filter and insert your firewall endpoints into the traffic flow. Update the route tables for both directions of traffic flow, if you want to filter incoming and outgoing traffic.

For example, suppose you wanted to filter traffic that's currently routed between a customer subnet and an internet gateway. You would update your route table configuration as follows to insert a firewall endpoint into the traffic flow:

1. Change the customer subnet route table so that it directs internet-bound traffic to the firewall endpoint.
2. Change the internet gateway route table so that it directs traffic that's bound for the customer subnet to the firewall endpoint.
3. Create a route table for the firewall endpoint so that it directs internet-bound traffic to the internet gateway and directs traffic that's bound for any destination inside the VPC to the destination specification `local`.

In this way, the firewall endpoint sits between the customer subnet and the internet gateway and can filter all incoming and outgoing traffic for the customer subnet.

For an overview of common Network Firewall architectures, with example route table configurations, see Architecture and routing examples (p. 10).

For information about managing route tables for your VPC, see Route tables in the *Amazon Virtual Private Cloud User Guide*.

Transit gateway attachment configuration for AWS Network Firewall

This section applies to the use of Network Firewall with a transit gateway in multiple Availability Zones where the firewall endpoints might reside in different Availability Zones than the subnets whose traffic they're filtering.

> **Note**
> To use this configuration, you must enable appliance mode on the transit gateway VPC attachment for any VPC where Network Firewall endpoints reside.

A Network Firewall endpoint is a stateful network appliance. Enabling appliance mode ensures that the transit gateway continues to use the same Availability Zone for the VPC attachment over the lifetime of a flow of traffic between source and destination.

For information about VPC transit gateways, see the guide Amazon Virtual Private Cloud Transit Gateways.

For information about appliance mode and how to enable it in your attachments, see Availability Zones and Example: Appliance in a shared services VPC.

Firewalls in AWS Network Firewall

An AWS Network Firewall *firewall* connects a firewall policy, which defines network traffic monitoring and filtering behavior, to the VPC that you want to protect. The firewall configuration includes specifications for the Availability Zones and subnets where the firewall endpoints are placed. It also defines high-level settings like the firewall logging configuration and tagging on the AWS firewall resource.

Topics

Firewall settings

A firewall has the following top-level settings.

- **Name** – The identifier for the firewall. You assign a unique name to every firewall. You can't change the name of a firewall after you create it.
- **Description** – Optional additional information about the firewall. Fill in any information that might help you remember the purpose of the firewall and how you want to use it. The description is included in firewall lists in the console and through the APIs.
- **VPC** – The VPC that's associated with the firewall. This is the VPC that the firewall provides protection for.
- **Subnets** – The subnets to use for your firewall endpoints. You can specify up to one subnet for each Availability Zone that your VPC spans. See Configuring your VPC and other components for AWS Network Firewall (p. 27).
- **Firewall policy** – The firewall policy that's associated with the firewall. The firewall policy provides the monitoring and protection behavior for the firewall. You can use the same firewall policy for more than one firewall. For more information about firewall policies, see Firewall policies in AWS Network Firewall (p. 34).
- **Logging** – The type and location of the logs that Network Firewall provides for the firewall's stateful rules engine. You can enable flow logging for the network traffic that passes through the stateful rules engine. You can also enable alert logging for traffic that matches the stateful rules that have an action setting of `Alert` or `Drop`. For more information, see Logging network traffic from AWS Network Firewall (p. 83), and Stateful actions (p. 55).
- **Tags** – Zero or more key-value tag pairs. A tag is a label that you assign to an AWS resource. You can use tags to search and filter your resources and to track your AWS costs. For more information, see Tagging AWS Network Firewall resources (p. 99).
- **Delete protection** – A Boolean setting that is enabled when you create a firewall, and protects against accidental deletion of the firewall. The setting isn't shown in the console because the firewall deletion process disables this protection. Through the API, you must explicitly disable delete protection before you can delete the firewall.

Managing your firewall in AWS Network Firewall

This section describes how to create, update, and delete your firewall in AWS Network Firewall.

How Network Firewall propagates your changes

When you make any changes to a firewall, including changes to any of the firewall's components, like rule groups and firewall policies, Network Firewall propagates the changes everywhere that the firewall is 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. For example, if you modify a rule group so that it drops an additional type of packet, for a firewall that uses the rule group, the new packet type might briefly be dropped by one firewall endpoint while still being allowed by another.

This temporary inconsistency can occur when you first create a firewall and when you make changes to an existing firewall. Generally, any inconsistencies of this type last only a few seconds.

When you update rules in a stateful rule group and the updates don't change the rule order, Network Firewall propagates the new rules without stopping and restarting the service. This minimizes service disruption for traffic flows that are already established. If the update does change from one rule order to another, the existing flows are still disrupted.

Changes to stateful rules are applied only to new traffic flows. Other firewall changes, including changes to stateless rules, are applied to all network packets.

Topics

Creating a firewall

To follow this procedure, the VPC that you want to protect must have at least one subnet available to host a firewall endpoint. For information, see VPC subnets (p. 27).

To create a firewall through the console

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Firewalls**.
3. Choose **Create firewall**.
4. Enter a **Name** to identify this firewall.

   Note
   You can't change the name after you create the firewall.
5. (Optional) Enter a **Description** for the firewall.
6. Choose your **VPC** from the dropdown list.

   Note
   You can't change the VPC after you create the firewall.
7. For **Firewall subnets**, choose the Availability Zones and subnets that you want to use for your firewall endpoints. You can choose up to one subnet for each Availability Zone that your VPC spans. The subnets should be dedicated for Network Firewall firewall use. For more information, see VPC subnets (p. 27).
8. For the **Associated firewall policy** section, choose the firewall policy that you want to associate with the firewall. If you already have a firewall policy defined, you can select it. Otherwise, you can associate an empty policy, which you must name permanently here. If you associate an empty policy, Network Firewall creates the policy and you can define its rules and other settings using the procedure at Updating a firewall policy (p. 37).
9. (Optional) For the **Firewall tags - *optional*** section, assign key-value tags to your firewall. For information about tagging your AWS resources, see Tagging AWS Network Firewall resources (p. 99).

10. Choose **Create firewall**.

Your new firewall is added to the list in the **Firewalls** page.

Perform the following additional steps to finish configuring your new firewall and start using it to filter your network traffic.

- Configure the associated firewall policy, if it's not configured already. For information, see Firewall policies in AWS Network Firewall (p. 34).
- Optionally configure logging for your firewall. For information, see Logging network traffic from AWS Network Firewall (p. 83).
- Configure your VPC route tables to send traffic through the firewall endpoints. For information, see VPC route table configuration for AWS Network Firewall (p. 28).

Updating a firewall

To make changes to your firewall settings through the console, use the following procedure:

> **Warning**
> If your firewall update changes your stateful rule evaluation order type, you will experience an interruption of in-flight traffic through the firewall for a few seconds during the reset. This is the only type of update that has this effect. For more information about stateful rule evaluation order types, see Evaluation order for stateful rule groups (p. 42).

To update a firewall

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Firewalls**.
3. In the **Firewalls** page, choose the name of the firewall that you want to edit. This takes you to the firewall's details page.
4. Choose the tab **Firewall details**, then, in each section where you want to make changes, choose **Edit** and follow the console guidance to make your changes.
   - In the **Details** section, you can change the firewall description. The name is fixed after creation.
   - In the **Associated policy and VPC** section, you can add and remove Availability Zones and subnets and you can associate a different firewall policy. The VPC is fixed after creation.
   - In the **Logging** section, you can change your logging configuration for alert and flow logs. For information about your logging options, see Logging network traffic from AWS Network Firewall (p. 83).
   - In the **Firewall tags** section, you can change the tags assigned to the AWS firewall resource. For information about tagging, see Tagging AWS Network Firewall resources (p. 99).
5. Choose **Save** to save your changes and return to the firewall's detail page.

Deleting a firewall

To delete a firewall through the console, you first disassociate all AWS resources from the firewall. Perform the following procedure.

To delete a firewall

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.

2. In the navigation pane, under **Network Firewall**, choose **Firewalls**.
3. In the **Firewalls** page, select the firewall that you want to delete.
4. Choose **Delete**, and then confirm your request.

Your firewall is removed from the list in the **Firewalls** page. The removal can take a few minutes to complete.

Firewall policies in AWS Network Firewall

An AWS Network Firewall *firewall policy* defines the monitoring and protection behavior for a firewall. The details of the behavior are defined in the rule groups that you add to your policy, and in some policy default settings. To use a firewall policy, you associate it with one or more firewalls.

Topics

Firewall policy settings

A firewall policy has the following top-level settings.

- **Name** – The identifier for the firewall policy. You assign a unique name to every firewall policy. You can't change the name of a firewall policy after you create it.
- **Description** – Optional additional information about the firewall policy. Fill in any information that might help you remember the purpose of the firewall policy and how you want to use it. The description is included in firewall policy lists in the console and through the APIs.
- **Stateless rule groups** – Zero or more collections of stateless rules, with priority settings that define their processing order within the policy. For information about creating and managing rule groups for use in your policies, see Rule groups in AWS Network Firewall (p. 39).
- **Stateless default actions** – Define how Network Firewall handles a packet or UDP packet fragment that doesn't match any of the rules in the stateless rule groups. Network Firewall silently drops packet fragments for other protocols. The options for the firewall policy's default settings are the same as for stateless rules. For more information about the options, see Stateless default actions in your firewall policy (p. 35).
- **Stateful engine options** – The structure that holds stateful rule order settings. Note that you can only configure RuleOrder settings when you first create the policy. RuleOrder can't be edited later.
- **Stateful rule groups** – Zero or more collections of stateful rules, provided in Suricata compatible format. For information about creating and managing rule groups for use in your policies, see Rule groups in AWS Network Firewall (p. 39).
- **Stateful default actions** – Define how Network Firewall handles a packet that doesn't match any of the rules in the stateful rule groups. For more information about the options, see Stateful default actions in your firewall policy (p. 35).
- **Tags** – Zero or more key-value tag pairs. A tag is a label that you assign to an AWS resource. You can use tags to search and filter your resources and to track your AWS costs. For more information about tags, see Tagging AWS Network Firewall resources (p. 99).

Capacity limitations

Network Firewall uses capacity calculations and limiting to control the operating resources that are required to process your rule groups and firewall policies. Each rule group has a capacity setting

that's reserved for it in the firewall policy when you add it. Additionally, the firewall policy has limits on the count of rule groups that you can add. For information about limits, see *Network Firewall quotas* (p. 101) for information about rule group capacity, see the section called "Rule group capacity" (p. 54).

Stateless default actions in your firewall policy

In your firewall policy configuration, you indicate how Network Firewall should handle packets that don't match any stateless rule group that's defined for the policy. You provide this configuration regardless of whether you define stateless rule groups for the policy.

The firewall policy allows you to specify different default settings for full packets and for UDP packet fragments. Network Firewall silently drops packet fragments for other protocols. The action options are the same as for the stateless rules that you use in the firewall policy's stateless rule groups.

You are required to specify one of the following options:

- **Pass** – Discontinue all inspection of the packet and permit it to go to its intended destination.
- **Drop** – Discontinue all inspection of the packet and block it from going to its intended destination.
- **Forward to stateful rules** – Discontinue stateless inspection of the packet and forward it to the stateful rule engine for inspection.

Additionally, you can optionally specify a named custom action to apply. For this action, Network Firewall assigns a dimension to Amazon CloudWatch metrics with the name set to `CustomAction` and a value that you specify. For more information, see AWS Network Firewall metrics in Amazon CloudWatch (p. 97).

After you define a named custom action, you can use it by name in the same context as where you defined it. You can reuse a custom action setting among the rules in a rule group and you can reuse a custom action setting between the two default stateless custom action settings for a firewall policy.

Stateful default actions in your firewall policy

In your firewall policy configuration, you indicate how Network Firewall should handle packets that don't match any stateful rule group that's defined for the policy. Note that you can provide this configuration regardless of whether you define stateful rule groups for the policy when you use strict rule order.

For more information about stateful default actions for rule groups, see Default action order (p. 42).

Managing your firewall policy in AWS Network Firewall

This section describes how to create, update, and delete your firewall policy in Network Firewall.

How Network Firewall propagates your changes

When you make any changes to a firewall, including changes to any of the firewall's components, like rule groups and firewall policies, Network Firewall propagates the changes everywhere that the firewall is 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. For example, if you modify a rule group so

that it drops an additional type of packet, for a firewall that uses the rule group, the new packet type might briefly be dropped by one firewall endpoint while still being allowed by another.

This temporary inconsistency can occur when you first create a firewall and when you make changes to an existing firewall. Generally, any inconsistencies of this type last only a few seconds.

When you update rules in a stateful rule group and the updates don't change the rule order, Network Firewall propagates the new rules without stopping and restarting the service. This minimizes service disruption for traffic flows that are already established. If the update does change from one rule order to another, the existing flows are still disrupted.

Changes to stateful rules are applied only to new traffic flows. Other firewall changes, including changes to stateless rules, are applied to all network packets.

Topics

Creating a firewall policy

To create a firewall policy, you need rule groups that you've already defined to use in the policy. You can create new rule groups and reuse existing ones. For information about creating and managing rule groups, see Rule groups in AWS Network Firewall (p. 39).

To create a firewall policy

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Firewall policies**.
3. Choose **Create firewall policy**.
4. Enter a **Name** to identify this firewall policy.

   Note
   You can't change the name after you create the firewall policy.
5. (Optional) Enter a **Description** for the policy.
6. Choose **Next** to go to the firewall policy's **Add rule groups** page.
7. To choose the actions to take on packets that don't match any stateless rules, in the **Stateless default actions** section, first choose how to treat fragmented packets. You can choose **Use the same actions for all packets** or **Use different actions for full packets and fragmented packets**. You can then choose **Pass**, **Drop**, or **Forward to stateful rule groups** for all packets, or choose individually for full and fragmented packets. You also have the option to enable a custom action that lets you publish Amazon CloudWatch metrics and their dimension values.
8. To choose the way that your stateful rules are ordered for evaluation, and the actions to take on packets that don't match any stateful rules, in the **Stateful rule order and default action** section, first choose a rule order:

   - Choose **Default** to have the stateful rules engine determine the evaluation order of your rules. The default action for this rule order is **Drop**, which drops packets that don't match any rules.
   - Choose **Strict** to provide your rules in the order that you want them to be evaluated. You can then choose one or more default actions for packets that don't match any rules.

   For more information about stateful default actions for rule groups, see Default action order (p. 42).

9. To add stateless rule groups, in the **Stateless rule groups** section, choose **Add rule groups**, then select the check boxes for the rule groups that you want to add and choose **Add rule groups**.
10. If your firewall policy has multiple stateless rule groups, in the **Stateless rule group** section, update the processing order as needed. Network Firewall processes stateless rule groups by order of priority, starting from the lowest. To move a rule group in the list, select the check box next to its name and then move it up or down. For more information, see How AWS Network Firewall filters network traffic (p. 9).
11. Choose the stateless default actions for the firewall policy to take if a packet or UDP packet fragment doesn't match any of the stateless rule groups. Network Firewall silently drops packet fragments for other protocols. For information about the action options, see Stateless default actions in your firewall policy (p. 35).

    Network Firewall doesn't automatically forward packets to stateful rule groups. It forwards only for the following situations:

    - The packet matches a stateless rule whose action specifies forward to stateful rule groups.
    - The packet doesn't match any stateless rule and the applicable default action setting specifies forward to stateful rule groups.
12. To add stateful rule groups, in the **Stateful rule groups** section, choose **Add rule groups**, then select the check boxes for the rule groups that you want to add and choose **Add rule groups**.
13. Choose **Next**.
14. (Optional) On the **Add tags** page, enter a key and optional value for any tag that you want added to this firewall policy. Tags help you organize and manage your AWS resources. For more information about tagging your resources, see Tagging AWS Network Firewall resources (p. 99).
15. Choose **Next**.
16. In the **Review and create** page, check over your firewall policy settings. If you want to change any section, choose **Edit** for the section. This returns you to the page in the firewall policy wizard. Make your changes, then choose **Next** on each page until you come back to the review and create page.
17. Choose **Create firewall policy**.

Your new firewall policy is added to the list in the **Firewall policies** page.

Updating a firewall policy

To change your firewall policy settings, use the following procedure:

To update a firewall policy

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Firewall policies**.
3. In the **Firewall policies** page, select the name of the firewall policy you want to update.
4. In the firewall policy's page, make your changes. You can't change the name of a firewall policy after creation, but you can change other details and you can change the rule groups.
5. Choose **Save** to save your changes.

How Network Firewall propagates your changes

When you make any changes to a firewall, including changes to any of the firewall's components, like rule groups and firewall policies, Network Firewall propagates the changes everywhere that the firewall is 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. For example, if you modify a rule group so

that it drops an additional type of packet, for a firewall that uses the rule group, the new packet type might briefly be dropped by one firewall endpoint while still being allowed by another.

This temporary inconsistency can occur when you first create a firewall and when you make changes to an existing firewall. Generally, any inconsistencies of this type last only a few seconds.

When you update rules in a stateful rule group and the updates don't change the rule order, Network Firewall propagates the new rules without stopping and restarting the service. This minimizes service disruption for traffic flows that are already established. If the update does change from one rule order to another, the existing flows are still disrupted.

Changes to stateful rules are applied only to new traffic flows. Other firewall changes, including changes to stateless rules, are applied to all network packets.

Deleting a firewall policy

To delete a firewall policy, perform the following procedure.

Deleting a rule group or firewall policy

When you delete a rule group or a firewall policy, AWS Network Firewall checks to see if it's currently being referenced. A rule group can be referenced by a firewall policy, and a firewall policy can be referenced by a firewall. If Network Firewall determines that the resource is being referenced, it warns you. Network Firewall is almost always able to determine whether a resource is being referenced. However, in rare cases, it might not be able to do so. If you need to be sure that the resource that you want to delete isn't in use, check all of your firewalls or firewall policies before deleting it. Note that policies that have associations can't be deleted.

To delete a firewall policy

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Firewall policies**.
3. In the **Firewall policies** page, select firewall policy that you want to delete.
4. Choose **Delete**, and confirm your request.

Your firewall policy is removed from the list in the **Firewall policies** page.

Rule groups in AWS Network Firewall

An AWS Network Firewall *rule group* is a reusable set of criteria for inspecting and handling network traffic. You add one or more rule groups to a firewall policy as part of policy configuration. For more information about firewall policies and firewalls, see Firewall policies in AWS Network Firewall (p. 34) and Firewalls in AWS Network Firewall (p. 30).

Network Firewall rule groups are either *stateless* or *stateful*. Stateless rule groups evaluate packets in isolation, while stateful rule groups evaluate them in the context of their traffic flow. You can create and manage the following categories of rule groups in Network Firewall:

- **Stateless** – Defines standard, 5-tuple criteria for examining a packet on its own, with no additional context.
- **Stateful** – Defines criteria for examining a packet in the context of traffic flow and of other traffic that's related to the packet.

  Network Firewall uses a Suricata rules engine to process all stateful rules. You can write any of your stateful rules in Suricata compatible format. Alternately, for domain list rules and for very basic rules, you can use an easy entry form provided by Network Firewall.

  Stateful rule groups are available in the following categories:
  - **Suricata compatible IPS rules** – Defines intrusion prevention system (IPS) rules in the rule group, in Suricata compatible format. You can provide all of your stateful rules through this method if you want to.
  - **Domain list entry** – Defines a list of domain names and specifies the protocol type to inspect.
  - **Simple rules entry** – Defines standard, 5-tuple criteria for examining a packet within the context of a traffic flow. This entry method is available with all of its options only through the API.

Depending on the type of rule group, you might also define rules inside the rule group. Rules provide detailed criteria for packet inspection and specify what to do when a packet matches the criteria. When Network Firewall finds a match between the criteria and a packet, we say that the packet matches the rule group.

This section provides guidance for creating and managing your rule groups.

Topics

Common rule group settings in AWS Network Firewall

Every rule group has the following top-level settings:

- **Type** – Whether the rule group is stateless or stateful.
- **Name** – Identifier for the rule group. You assign a unique name to every rule group. You can't change the name of a rule group after you create it.
- **Description** – Optional additional information about the rule group. Fill in any information that might help you remember the purpose of the rule group and how you want to use it. The description is included in rule group lists in the console and through the APIs.
- **Capacity** – Limit on the processing requirements for the rule group. You can't change this setting after you create the rule group. For more information, including how to estimate your required capacity for a rule group, see Rule group capacity in AWS Network Firewall (p. 54).
- **Rules** – Set of packet inspection criteria used in the rule group. Rules in a rule group are either stateless or stateful, depending on the rule group type.
- **Tags** – Zero or more key-value tag pairs. A tag is a label that you assign to an AWS resource. You can use tags to search and filter your resources and to track your AWS costs. For more information, see Tagging AWS Network Firewall resources (p. 99).

Stateless rule groups in AWS Network Firewall

For stateless rule groups, the AWS Network Firewall stateless rules engine examines each packet in isolation. Network Firewall doesn't consider context such as traffic direction or other related packets.

Network Firewall supports the standard stateless 5-tuple rule specification for network traffic inspection. When Network Firewall finds a match between a rule's inspection criteria and a packet, we say that the packet matches the rule and its rule group, and Network Firewall applies the rule's specified action to the packet.

You can add multiple stateless rules to your stateless rule group.

All rule groups have the common settings that are defined at the section called “Common rule group settings” (p. 39).

General settings

A stateless rule has the following general settings.

- **Priority** – Number that indicates the processing order of the stateless rule within the rule group. This must be unique within the stateless rule group and it must be a positive integer. Network Firewall processes the rules starting from the lowest numbered priority setting. When you plan the rules in your rule group, provide priority settings with space in between, to leave yourself room to add rules later. For example, you might start by using priority settings that are multiples of 100.
- **Actions** – Defines how Network Firewall handles a packet that matches the rule match settings. You assign one standard setting, from among pass, drop, and forward to stateful. You can optionally add a custom setting, for example, to send metrics for the rule match to Amazon CloudWatch metrics. For more information about actions, see Rule actions in AWS Network Firewall (p. 55).

Match settings

A stateless rule has the following match settings. These specify what the Network Firewall stateless rules engine looks for in a packet. To be a match, a packet must satisfy all of the match settings in the rule.

- **Protocol** – Valid settings include `ALL` and specific protocol settings, like `UDP` and `TCP`. You can choose more than one specific setting.
- **Source** – Source IP addresses and ranges. If specified, a packet must come from a source address that's included in this list in order to match.

- **Source port range** – Source ports and port ranges. If specified, a packet must have a source port that's included in this list in order to match.
- **Destination** – Destination IP addresses and ranges. If specified, a packet must have a destination address that's included in this list in order to match.
- **Destination port range** – Destination ports and port ranges. If specified, a packet must have a destination port that's included in this list in order to match.
- **Optional TCP flags** – Optional, standard TCP flag settings, which indicate which flags to inspect and the values to inspect for. Each flag can be either enabled or disabled. You indicate the flags that you want to inspect in a masks setting, and then you indicate which of those flags must be enabled in the flags setting in order to match. The flags that you specify in the masks setting and don't specify in the flags setting must be unset in order to match.

Example

To create a very simple stateless rule group that passes all traffic from two CIDR blocks, you could provide the following stateless rule settings in a single rule:

- **Priority** – `100`
- **Action** – `PASS`
- **Protocol** – `ALL`
- **Source** – `192.0.2.0/8`, `198.51.100.0/16`

To block all other traffic, you would set the firewall policy's stateless default actions to `Drop`. For more information, see Stateless default actions in your firewall policy (p. 35).

Stateful rule groups in AWS Network Firewall

A stateful rule group is a rule group that uses Suricata compatible intrusion prevention system (IPS) specifications. Suricata is an open source network IPS that includes a standard rule-based language for stateful network traffic inspection. AWS Network Firewall supports Suricata version 5.0.2.

Stateful rule groups have a configurable top-level setting called `StatefulRuleOptions`, which contains the `RuleOrder` attribute. You can set this in the console when you create a rule group, or in the API under `StatefulRuleOptions`. You can't change the `RuleOrder` after the rule group is created.

You can enter any stateful rule in Suricata compatible strings. For standard Suricata rules specifications and for domain list inspection, you can alternately provide specifications to Network Firewall and have Network Firewall create the Suricata compatible strings for you.

As needed, depending on the rules that you provide, the stateful engine performs deep packet inspection (DPI) of your traffic flows. DPI inspects and processes the payload data within your packets, rather than just the header information.

The rest of this section provides requirements and additional information for using Suricata compatible rules with Network Firewall. For full information about Suricata, see the Suricata website at Suricata and the Suricata User Guide. AWS Network Firewall supports Suricata version 5.0.2.

Topics

- Examples of stateful rules for Network Firewall (p. 49)

Limitations and caveats for stateful rules in AWS Network Firewall

AWS Network Firewall stateful rules are Suricata compatible. Most Suricata rules work out of the box with Network Firewall. Your use of Suricata rules with Network Firewall has the restrictions and caveats listed in this section.

Not supported

The following Suricata features are not supported by Network Firewall:

- IP reputation. The `iprep` keyword is not allowed.
- Lua scripting.
- GeoIP.
- File extraction. File keywords aren't allowed.
- Thresholding.
- ENIP/CIP keywords.
- Datasets. The keywords `dataset` and `datarep` aren't allowed.
- Rules actions except for pass, drop, and alert. Pass, drop, and alert are supported. For additional information about stateful rule actions, see Stateful actions (p. 55).

Supported with caveats

The following Suricata features have caveats for use with Network Firewall:

- To create a rule that requires a variable, you must specify the variable in the rule group. Without the required variables, the rule group isn't valid. For an example of a rule group that's configured with variables, see Rule with variables (p. 49).
- In payload keywords, the `pcre` keyword is only allowed with the `content` keyword.
- The `priority` keyword is not supported for rule groups that evaluate rules using strict evaluation order.

Evaluation order for stateful rule groups

All of your stateful rule groups are provided to the rule engine as Suricata compatible strings. Suricata can evaluate stateful rule groups by using the default rule group ordering method, or you can set an exact order using the *strict* ordering method. The settings for your rule groups must match the settings for the firewall policy that they belong to.

Default action order

If your firewall policy is set up to use default rule group ordering, the default action order by which Suricata evaluates stateful rules is determined by the following settings, listed in order of precedence:

1. The Suricata `action` specification. This takes highest precedence.

   Actions are processed in the following order:

   a. `pass`

   b. `drop`
   c. `alert`

   For more information about the action specification, see Suricata.yaml: Action-order in the Suricata User Guide.
2. The Suricata `priority` keyword. Within a specific action group, you can use the priority setting to indicate the processing order. By default, Suricata processes from the lowest numbered priority setting on up. The `priority` keyword has a mandatory numeric value ranging from 1 to 255. Note that the `priority` keyword is only valid using the default action order.

   For more information about priority, see Suricata.yaml: Action-order in the Suricata User Guide.

For example, Suricata evaluates all `pass` rules before evaluating any `drop` or `alert` rules by default, regardless of the value of priority settings. Within all `pass` rules, if priority keywords are present, Suricata orders the processing according to them.

The protocol layer does not impact the rule evaluation order by default. If you want to avoid matching against lower-level protocol packets before higher-level application protocols can be identified, consider using the `flow` keyword in your rules. This is needed because, for example, a TCP rule might match on the first packet of a TCP handshake before the stateful engine can identify the application protocol. For information about the `flow` keyword, see Flow Keywords.

For examples of default rule order management, see Managing rule evaluation order (p. 52).

For additional information about evaluation order for stateful rules, see the following topics in the Suricata User Guide:

- Suricata.yaml: Action-order
- Meta Keywords: priority

Strict evaluation order

If your firewall policy is set up to use strict ordering, Network Firewall allows you the option to manually set a *strict* rule group order. With strict ordering, the rule groups are evaluated by order of priority, starting from the lowest number, and the rules in each rule group are processed in the order in which they're defined.-->

When you choose **Strict** for your rule order, you can choose one or more **Default actions**. Note that this does not refer to default action rule ordering, but rather, to the default actions that Network Firewall takes when following your strict, or exact, rule ordering. The default actions are as follows:

Drop actions:

Choose none or one. You can't choose both.

- **Drop all** – Drops all packets.
- **Drop established** – Drops only the packets that are in established connections. This allows the layer 3 and 4 connection establishment packets that are needed for the upper-layer connections to be established, while dropping the packets for connections that are already established. This allows application-layer *pass* rules to be written in a default-deny setup without the need to write additional rules to allow the lower-layer handshaking parts of the underlying protocols.

Alert actions:

Choose none, one, or both.

- **Alert all** - Logs an `ALERT` message on all packets. This does not drop packets, but alerts you to what would be dropped if you were to choose **Drop all**.
- **Alert established** - Logs an `ALERT` message on only the packets that are in established connections. This does not drop packets, but alerts you to what would be dropped if you were to choose **Drop established**.

For more information about logging network traffic, see Logging network traffic from AWS Network Firewall (p. 83).

How to provide stateful rules to AWS Network Firewall

When you create a stateful rule group from Suricata compatible rules, you can provide the rules to the rule group creation operation in one of the following ways:

- Rule strings that are written in Suricata compatible syntax. When you use this option, Network Firewall passes your rule strings to Suricata for processing.
- Domain list rule specification. With this option, Network Firewall translates your rule specification into Suricata compatible rules and then passes the resulting rule strings to Suricata for processing.
- Standard, simple rule group specification. With this option, Network Firewall translates your specification into Suricata compatible rules and then passes the resulting rule strings to Suricata for processing.

The sections that follow provide details for each of these options.

Suricata compatible rule strings in AWS Network Firewall

All rule groups have the common settings that are defined at the section called “Common rule group settings” (p. 39).

For this rule group type, you provide match and action settings in a string, in a Suricata compatible specification. Your specification fully defines what the stateful rules engine looks for in a traffic flow and the action to take on the packets in a flow that matches the inspection criteria.

You can provide your Suricata compatible specification to Network Firewall in rules strings or files, depending on how you're accessing Network Firewall.

- **Console** – In the AWS Management Console, provide the rules string in the text box that appears for the stateful rule group option **Import Suricata compatible rules**. For information about using the console to manage your rule group, see Creating a stateful rule group (p. 57).
- **API** – Through the API, you can provide either the rules or the name of the file that contains the rules. In a file, Suricata compatible rules are usually written one rule per line.

  You provide either the file or the rules string in the `RulesString` field within the `RuleGroup` structure when you create or update the rule group. For information, see CreateRuleGroup in the *AWS Network Firewall API Reference*.
- **CLI** – Through the CLI, you can provide the rules, the name of a file that contains the rules, or the name of a file that contains the rule group structure in JSON format, with the rules defined in that.

  The following listing shows the syntax for providing the rules in a file. To use a command like this, substitute in your new rule group name, its calculated capacity, and the JSON rules file name.

```
aws network-firewall create-rule-group --rule-group-name <ruleGroupName> --capacity
 <capacityCalculation> --type STATEFUL --rules <rules file name>
```

Stateful domain list rule groups in AWS Network Firewall

AWS Network Firewall supports domain name stateful network traffic inspection. You can create allow lists and deny lists with domain names that the stateful rules engine looks for in network traffic.

All rule groups have the common settings that are defined at the section called “Common rule group settings” (p. 39).

General settings

A domain list rule group has the following general settings.

- **Action** – Defines whether Network Firewall allows traffic that matches the rule match settings. Valid values for domain rules are `Allow` and `Deny`. For `Allow`, traffic of the specified protocol type that doesn't match the domain specifications is denied. For more information about actions, see Rule actions in AWS Network Firewall (p. 55).
- **(Optional) `HOME_NET` rule group variable** – Used to expand the local network definition beyond the CIDR range of the VPC where you deploy Network Firewall. You can only set this outside of the console. For more information, see Domain list inspection for traffic from outside the deployment VPC (p. 45).

Match settings

A domain list rule group has the following match settings. These specify what the Network Firewall stateful rules engine looks for in a packet. A packet must satisfy all match settings to be a match.

- **Domain list** – List of strings specifying the domain names that you want to match. A packet must match one of the domain specifications in the list to be a match for the rule group. Valid domain name specifications are the following:
  - Explicit names. For example, `abc.example.com` matches only the domain `abc.example.com`.
  - Names that use a domain wildcard, which you indicate with an initial `'.'`. For example, `.example.com` matches `example.com` and matches all subdomains of `example.com`, such as `abc.example.com` and `www.example.com`.
- **Protocols** – You can inspect HTTP or HTTPS protocols, or both.

For HTTPS traffic, Network Firewall uses the Server Name Indication (SNI) extension in the TLS handshake to determine the hostname, or domain name, that the client is trying to connect to. For HTTP traffic, Network Firewall uses the HTTP host header to get the name. In both cases, Network Firewall doesn't pause connections to do out-of-band DNS lookups. It uses the SNI or host header, not the IP addresses, when evaluating domain list rule groups. If you want to inspect IP addresses, to mitigate situations where the SNI or host headers have been manipulated, write separate rules for that and use them in conjunction with or in place of your domain list rules.

For examples of domain list specifications and the Suricata compatible rules that Network Firewall generates from them, see Domain filtering (p. 50).

Domain list inspection for traffic from outside the deployment VPC

To use domain name filtering for traffic from outside the VPC where you've deployed Network Firewall, you must manually set the `HOME_NET` variable for the rule group. The most common use case for this is a central firewall VPC with traffic coming from other VPCs through a transit gateway.

> **Note**
> You can't set a rule group variable through the console. If you usually work through the console, you can create your rule group through the console, then use the command line to provide the variable setting. The guidance that follows shows how to do the command line work.

By default, domain list inspection uses a `HOME_NET` that is set to the CIDR range of the VPC where Network Firewall is deployed. Only traffic from that range is passed through the domain list filtering. To filter traffic from outside the deployment VPC, you must provide a `HOME_NET` setting that includes the other CIDR ranges that you want to inspect, along with the CIDR range of the VPC where Network Firewall is deployed.

For example, say that the VPC where you deploy Network Firewall has the CIDR range `192.0.2.0/24`. In addition to the traffic for that VPC, you want to filter traffic for two other VPCs that have CIDR ranges `10.0.0.0/16` and `10.1.0.0/16`. You're using a domain list rule group named `domains`.

The following command line call retrieves the JSON listing for the rule group:

```
aws network-firewall describe-rule-group --type STATEFUL \
--rule-group-name domains --region us-west-2
```

The following shows the example JSON response. This rule group has only `RulesSource` defined, which contains the domain list inspection specifications.

```
{
    "UpdateToken": "a4648a25-e315-4d17-8553-283c2eb33118",
    "RuleGroup": {
        "RulesSource": {
            "RulesSourceList": {
                "Targets": [
                    ".example.com",
                    "www.example.org"
                ],
                "TargetTypes": [
                    "HTTP_HOST",
                    "TLS_SNI"
                ],
                "GeneratedRulesType": "DENYLIST"
            }
        }
    },
    "RuleGroupResponse": {
        "RuleGroupArn": "arn:aws:network-firewall:us-west-2:111122223333:stateful-
rulegroup/domains",
        "RuleGroupName": "domains",
        "RuleGroupId": "f3333333-fb99-11c1-bbe3-1d1caf1d1111",
        "Type": "STATEFUL",
        "Capacity": 100,
        "RuleGroupStatus": "ACTIVE",
        "Tags": []
    }
}
```

Variable settings are defined for a rule group in a `RuleVariables` setting. This rule group currently has no `HOME_NET` variable declaration, so we know that `HOME_NET` is set to the default. In our example case, it's `192.0.2.0/24`.

To add CIDR ranges to the `HOME_NET` setting, we update the rule group with our variable declaration. The following shows a file named `variables.json` that contains the rule group JSON with the added variables settings:

```
{
    "RuleVariables": {
        "IPSets": {
           "HOME_NET": {
             "Definition": [
               "10.0.0.0/16",
```

```
                "10.1.0.0/16",
                "192.0.2.0/24"
              ]
            }
        }
    },
    "RulesSource": {
        "RulesSourceList": {
           "Targets": [
                ".example.com",
                "www.example.org"
           ],
           "TargetTypes": [
                "HTTP_HOST",
                "TLS_SNI"
           ],
           "GeneratedRulesType": "DENYLIST"
        }
    }
}
```

The following command uses the `variables.json` file to update the rule group definition with the correct `HOME_NET` settings:

```
aws network-firewall update-rule-group \
--rule-group-arn arn:aws:network-firewall:us-west-2:111122223333:stateful-rulegroup/domains
 \
--update-token a4648a25-e315-4d17-8553-283c2eb33118 \
--rule-group file://variables.json \
--region us-west-2
```

The following shows an example response to the call:

```
{
    "UpdateToken": "32ebfb82-40a2-4896-b34d-91dada978f67",
    "RuleGroupResponse": {
        "RuleGroupArn": "arn:aws:network-firewall:us-west-2:111122223333:stateful-
rulegroup/domains",
        "RuleGroupName": "domains",
        "RuleGroupId": "f3333333-fb99-11c1-bbe3-1d1caf1d1111",
        "Type": "STATEFUL",
        "Capacity": 100,
        "RuleGroupStatus": "ACTIVE",
        "Tags": []
    }
}
```

If we retrieve the `domains` rule group again, we see that the rule group has the added variable definition:

```
aws network-firewall describe-rule-group --type STATEFUL \
--rule-group-name domains --region us-west-2
```

The response JSON contains the added variable:

```
{
    "UpdateToken": "42ffac91-20b5-5512-a24c-85cbca797e23",
    "RuleGroup": {
        "RuleVariables": {
            "IPSets": {
```

```
                "HOME_NET": {
                    "Definition": [
                        "10.0.0.0/16",
                        "10.1.0.0/16",
                        "192.0.2.0/24"
                    ]
                }
            }
        },
        "RulesSource": {
            "RulesSourceList": {
                "Targets": [
                    ".example.com",
                    "www.example.org"
                ],
                "TargetTypes": [
                    "HTTP_HOST",
                    "TLS_SNI"
                ],
                "Generate470805dRulesType": "DENYLIST"
            }
        }
    },
    "RuleGroupResponse": {
        "RuleGroupArn": "arn:aws:network-firewall:us-west-2:111122223333:stateful-
rulegroup/domains",
        "RuleGroupName": "domains",
        "RuleGroupId": "f3333333-fb99-11c1-bbe3-1d1caf1d1111",
        "Type": "STATEFUL",
        "Capacity": 100,
        "RuleGroupStatus": "ACTIVE",
        "Tags": []
    }
}
```


Standard stateful rule groups in AWS Network Firewall

AWS Network Firewall supports easy entry for standard stateful rules for network traffic inspection. The match criteria for this stateful rule type is similar to the Network Firewall stateless rule.

All rule groups have the common settings that are defined at the section called “Common rule group settings” (p. 39).

General settings

A stateful basic rule has the following general settings.

- **Action** – Defines how Network Firewall handles a packet that matches the rule match settings. Valid values are pass, drop, and alert. For more information about actions, see Rule actions in AWS Network Firewall (p. 55).

Match settings

A basic stateful rule has the following match settings. These specify what the Network Firewall stateful rules engine looks for in a packet. To be a match, a packet must satisfy all of the match settings in the rule.

- **Protocol** – Transport protocol. Choose the protocol that you want to inspect. For all protocols, you can use `IP`, because all traffic on AWS and on the internet is IP.
- **Source** – Source IP addresses and ranges. If specified, a packet must come from a source address that's included in this list in order to match.

- **Source port** – Source ports and port ranges. If specified, a packet must have a source port that's included in this list in order to match.
- **Destination** – Destination IP addresses and ranges. If specified, a packet must have a destination address that's included in this list in order to match.
- **Destination port** – Destination ports and port ranges. If specified, a packet must have a destination port that's included in this list in order to match.
- **Traffic direction** – Direction of traffic flow. Valid settings are `Any` and `Forward`. `Forward` matches packets whose origination matches the rule's source settings and whose destination matches the rule's destination settings. `Any` matches the forward match, and also matches packets whose origination matches the rule's destination settings, and whose destination matches the rule's source settings.
- **Rule options** – Define the specifics of the rule, in keyword, settings pairs.

  Note
  The console doesn't currently allow entry of rule options. Rule options are usually required for complete specification of this rule type. If you need to specify rule options, use one of the APIs or AWS CloudFormation. For information, see StatefulRule in the *AWS Network Firewall API Reference* and AWS::NetworkFirewall::RuleGroup StatefulRule in the *AWS CloudFormation User Guide*.

For an example rule specification and the Suricata compatible rule that Network Firewall generates from it, see Standard stateful rule groups (p. 52).

Best practices for writing Suricata compatible rules for AWS Network Firewall

When you write your stateful rules, verify the configuration of the firewall policy where you intend to use them, to make sure that all of your rules evaluate as you want them to. For information about how AWS Network Firewall handles network traffic and when it sends it to the stateful engine, see How AWS Network Firewall filters network traffic (p. 9).

For example, many stateful rules rely on seeing a complete bidirectional traffic flow for correct evaluation, such as rules with options like `flow: established`. To use rules like this, you must configure your stateless rules and default actions to ensure forwarding of traffic for both directions to the stateful engine. For information about these action options, see Rule actions in AWS Network Firewall (p. 55) and Stateless default actions in your firewall policy (p. 35).

Examples of stateful rules for Network Firewall

This section lists examples of Suricata compatible rules that could be used with AWS Network Firewall.

Note
Examples are not intended to be used in your Network Firewall configuration exactly as they are listed.
The examples provide general information and sample rule specifications for common use cases. Before using any rule from these examples or elsewhere, test and adjust it carefully to be sure that it fits your needs. It's your responsibility to ensure that each rule that you use is suited to your specific use case and functioning the way that you want it to.

Rule with variables

Note
Before using any example rule listing, test and adapt it to your needs.

The following JSON defines an example Suricata compatible rule group that uses the variables `HTTP_SERVERS` and `HTTP_PORTS`, with the variable definitions provided in the rule group declaration.

```
{
"RuleVariables": {
    "IPSets": {
        "HTTP_SERVERS": {
            "Definition": [
                "10.0.2.0/24",
                "10.0.1.19"
            ]
        }
    },
    "PortSets": {
        "HTTP_PORTS": {
            "Definition": ["80", "8080"]
        }
    }
},
"RulesSource": {
    "RulesString": "alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:
\".htpasswd access attempt\"; flow:to_server,established; content:\".htpasswd\"; nocase;
 sid:210503; rev:1;)"
}
}
```

The variable `EXTERNAL_NET` is a Suricata standard variable that represents the traffic destination. For more information, see https://suricata.readthedocs.io/en/suricata-5.0.2/rules/intro.html#ports-source-and-destination.

Domain filtering

Note
Before using any example rule listing, test and adapt it to your needs.

Deny list example JSON, rule group creation, and generated Suricata rules

The following JSON shows an example rule definition for a Network Firewall domain list rule group that specifies a deny list.

```
{
    "RulesSource": {
        "RulesSourceList": {
            "Targets": [
                "evil.com"
            ],
            "TargetTypes": [
                 "TLS_SNI",
                 "HTTP_HOST"
             ],
             "GeneratedRulesType": "DENYLIST"
        }
    }
}
```

To use the Network Firewall rule specification, we save the JSON to a local file `domainblock.example.json`, and then create the rule group in the following CLI command:

```
aws network-firewall create-rule-group --rule-group-name "RuleGroupName" --type STATEFUL --
rule-group file://domainblock.example.json --capacity 1000
```

The following Suricata rules listing shows the rules that Network Firewall creates for the above deny list specification.

```
drop tls $HOME_NET any -> $EXTERNAL_NET any (tls.sni; content:"evil.com"; startswith;
 nocase; endswith; msg:"matching TLS denylisted FQDNs"; priority:1; flow:to_server,
 established; sid:1; rev:1; gid:255;)
drop http $HOME_NET any -> $EXTERNAL_NET any (http.host; content:"evil.com"; startswith;
 endswith; msg:"matching HTTP denylisted FQDNs"; priority:1; flow:to_server, established;
 sid:2; rev:1; gid:255;)
```

HTTP allow list example JSON and generated Suricata rules

The following JSON shows an example rule definition for a Network Firewall domain list rule group that specifies an HTTP allow list. The `.` before the domain name in `.amazon.com` is the wildcard indicator in Suricata.

```
{
    "RulesSource": {
        "RulesSourceList": {
            "Targets": [
                ".amazon.com",
                "example.com"
            ],
            "TargetTypes": [
                "HTTP_HOST"
            ],
            "GeneratedRulesType": "ALLOWLIST"
        }
    }
}
```

The following Suricata rules listing shows the rules that Network Firewall for the above allow list specification.

```
pass http $HOME_NET any -> $EXTERNAL_NET any (http.host; dotprefix; content:".amazon.com";
 endswith; msg:"matching HTTP allowlisted FQDNs"; priority:1; flow:to_server, established;
 sid:1; rev:1; gid:1024;)
pass http $HOME_NET any -> $EXTERNAL_NET any (http.host; content:"example.com"; startswith;
 endswith; msg:"matching HTTP allowlisted FQDNs"; priority:1; flow:to_server, established;
 sid:2; rev:1; gid:1024;)
drop http $HOME_NET any -> $EXTERNAL_NET any (http.header_names; content:"|0d 0a|";
 startswith; msg:"not matching any HTTP allowlisted FQDNs"; priority:1; flow:to_server,
 established; sid:3; rev:1; gid:1024;)
```

TLS allow list example JSON and generated Suricata rules

The following JSON shows an example rule definition for a Network Firewall domain list rule group that specifies a TLS allow list.

```
{
    "RulesSource": {
        "RulesSourceList": {
            "Targets": [
                ".amazon.com",
                "example.com"
            ],
            "TargetTypes": [
                "TLS_SNI"
            ],
            "GeneratedRulesType": "ALLOWLIST"
        }
    }
}
```

The following Suricata rules listing shows the rules that Network Firewall for the above allow list specification.

```
pass tls $HOME_NET any -> $EXTERNAL_NET any (tls.sni; dotprefix; content:".amazon.com";
 nocase; endswith; msg:"matching TLS allowlisted FQDNs"; priority:1; flow:to_server,
 established; sid:1; rev:1; gid:255;)
pass tls $HOME_NET any -> $EXTERNAL_NET any (tls.sni; content:"example.com"; startswith;
 nocase; endswith; msg:"matching TLS allowlisted FQDNs"; priority:1; flow:to_server,
 established; sid:2; rev:1; gid:255;)
drop tls $HOME_NET any -> $EXTERNAL_NET any (msg:"not matching any TLS allowlisted FQDNs";
 priority:1; flow:to_server, established; sid:3; rev:1; gid:255;)
```

Standard stateful rule groups

The following JSON shows an example rule definition for a Network Firewall basic stateful rule group.

```
{
    "RulesSource": {
        "StatefulRules": [
          {
            "Action": "DROP",
            "Header": {
                "Protocol": "HTTP",
                "Source": "$HOME_NET",
                "SourcePort": "ANY",
                "Direction": "ANY",
                "Destination": "$EXTERNAL_NET",
                "DestinationPort": "ANY"
            },
            "RuleOptions": [ {
                    "Keyword": "msg",
                    "Settings": [ "\"this is a stateful drop rule\""
                    ]
                },
                {
                    "Keyword": "sid",
                    "Settings": [ "1234"
                    ]
                }
            ]
        }
      ]
    }
}
```

The following Suricata rules listing shows the rules that Network Firewall generates for the above deny list specification.

```
drop http $HOME_NET ANY <> $EXTERNAL_NET ANY (msg:this is a stateful drop rule; sid:1234;
 gid:123;)
```

Managing rule evaluation order

Note
Before using any example rule listing, test and adapt it to your needs.

The examples in this section demonstrate ways to modify evaluation behavior by modifying rule evaluation order in Suricata compatible rules. For information about managing rule evaluation order, see Evaluation order for stateful rule groups (p. 42).

Allow HTTP traffic to specific domains only, allow all SSH traffic, and deny all other TCP traffic:

Default rule order method

```
drop tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg:"Drop established TCP:80"; flow:
 from_client,established; sid:172190; priority:5; rev:1;)
pass http $HOME_NET any -> $EXTERNAL_NET 80 (http.host; dotprefix; content:".example.com";
 endswith; msg:"Allowed HTTP domain"; priority:10; sid:172191; rev:1;)
pass tcp $HOME_NET any -> $EXTERNAL_NET 22 (msg:"Allow TCP 22"; sid:172192; rev:1;)
drop tcp $HOME_NET any -> $EXTERNAL_NET !80 (msg:"Drop All non-TCP:80";  sid:172193;
 priority:2; rev:1;)
```

Strict rule order method

```
pass http $HOME_NET any -> $EXTERNAL_NET 80 (http.host; dotprefix; content:".example.com";
 endswith; msg:"Allowed HTTP domain"; sid:172191; rev:1;)
pass tcp $HOME_NET any -> $EXTERNAL_NET 22 (msg:"Allow TCP 22"; sid:172192; rev:1;)
```

Allow HTTP traffic to specific domains only:

Default rule order method

```
pass http $HOME_NET any -> $EXTERNAL_NET 80 (http.host; dotprefix; content:".example.com";
 endswith; msg:"Allowed HTTP domain"; priority:1; sid:102120; rev:1;)
pass http $HOME_NET any -> $EXTERNAL_NET 80 (http.host; dotprefix;
 content:".mydomain.test"; endswith; msg:"Allowed HTTP domain"; priority:1; sid:102121;
 rev:1;)
drop http $HOME_NET any -> $EXTERNAL_NET 80 (msg:"Drop HTTP traffic"; priority:1;
 sid:102122; rev:1;)
```

Strict rule order method

```
pass http $HOME_NET any -> $EXTERNAL_NET 80 (http.host; dotprefix; content:".example.com";
 endswith; msg:"Allowed HTTP domain"; sid:102120; rev:1;)
pass http $HOME_NET any -> $EXTERNAL_NET 80 (http.host; dotprefix;
 content:".mydomain.test"; endswith; msg:"Allowed HTTP domain"; sid:102121; rev:1;)
```

Allow HTTP traffic to specific domains only and deny all other IP traffic:

Default rule order method

```
pass http $HOME_NET any -> $EXTERNAL_NET 80 (http.host; dotprefix; content:".example.com";
 endswith; msg:"Allowed HTTP domain"; priority:1; sid:892120; rev:1;)
drop tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg:"Drop established non-HTTP to TCP:80";
 flow: from_client,established; sid:892191; priority:5; rev:1;)
drop ip $HOME_NET any <> $EXTERNAL_NET any (msg: "Drop non-TCP traffic."; ip_proto:!
TCP;sid:892192; rev:1;)
drop tcp $HOME_NET any -> $EXTERNAL_NET !80 (msg:"Drop All non-TCP:80"; sid:892193;
 priority:2; rev:1;)
```

Strict rule order method

```
pass http $HOME_NET any -> $EXTERNAL_NET 80 (http.host; dotprefix; content:".example.com";
 endswith; msg:"Allowed HTTP domain"; sid:892120; rev:1;)
pass tcp $HOME_NET any <> $EXTERNAL_NET 80 (flow:not_established; sid:892191; rev:1;)
```

Rule group capacity in AWS Network Firewall

AWS Network Firewall uses capacity settings to calculate and manage the processing requirements for its rules groups and firewall policies. Each rule group must have a capacity setting that's fixed at creation. When you reference a rule group from a firewall policy, Network Firewall reserves the rule group's capacity in the policy, increasing the total capacity that's used by the policy.

Using the **consumed capacity** fields in the console, you can also describe a rule group or a policy to find out how much of the rule group or policy capacity is currently in use.

For information about the maximum capacity settings for rule groups and firewall policies, see AWS Network Firewall quotas (p. 101).

You can't change or exceed a rule group's capacity when you make changes to it, so when you set the rule group's capacity, leave room for it to grow.

Stateless rule group capacity

Estimate a stateless rule group's capacity as the sum of the capacities of the rules that you expect to have in it.

The capacity required for a single rule is the product of the complexity values of all of its match settings. For information about match settings, see Stateless rule groups in AWS Network Firewall (p. 40).

- A match setting with no criteria specified has a complexity value of 1. Through the console, the **All** and **Any** settings are equivalent to providing no criteria, and they have a complexity value of 1.
- A match setting with criteria specifications has a complexity value equal to the number of specifications in the setting. For example, a protocol specification set to `UDP` and a source specification set to `10.0.0.0/24` each have a value of 1. A protocol set to `UDP`, `TCP` has a value of 2 and a source set to `10.0.0.0/24`, `10.0.1.0/24`, `10.0.2.0/24` has a value of 3.

The following lists example calculations of stateless rule capacity requirements.

- A rule with protocol that specifies the two settings `UDP`, `TCP` and source with the three settings `10.0.0.0/24`, `10.0.1.0/24`, `10.0.2.0/24` and single or no specifications for the other match settings has a capacity requirement of 6.
- A rule with a protocol that specifies 30 different protocols, a source with 3 settings, and single or no specifications for the other match settings has a capacity requirement of 90.
- A rule with a protocol that specifies 30 different protocols, a source with 3 settings, a destination with 5 settings, and single or no specifications for the other match settings has a capacity requirement of (30*3*5) = 450.

To calculate the capacity of a rule group, add the capacity requirements of all rules that you expect to have in the rule group during its lifetime. You can't change this setting after you create the rule group.

The maximum capacity setting for a stateless rule group is 30,000.

Stateful rule group capacity

Estimate a stateful rule group's capacity as the number of rules that you expect to have in it during its lifetime. You can't change this setting after you create the rule group.

The maximum capacity setting for a stateful rule group is 30,000.

Rule actions in AWS Network Firewall

The rule action setting tells AWS Network Firewall how to handle a packet that matches the rule's match criteria.

Stateless actions

The action options for stateless rules are the same as for the firewall policy's default stateless rule actions.

You are required to specify one of the following options:

- **Pass** – Discontinue all inspection of the packet and permit it to go to its intended destination.
- **Drop** – Discontinue all inspection of the packet and block it from going to its intended destination.
- **Forward to stateful rules** – Discontinue stateless inspection of the packet and forward it to the stateful rule engine for inspection.

Additionally, you can optionally specify a named custom action to apply. For this action, Network Firewall assigns a dimension to Amazon CloudWatch metrics with the name set to `CustomAction` and a value that you specify. For more information, see AWS Network Firewall metrics in Amazon CloudWatch (p. 97).

After you define a named custom action, you can use it by name in the same context as where you defined it. You can reuse a custom action setting among the rules in a rule group and you can reuse a custom action setting between the two default stateless custom action settings for a firewall policy.

Stateful actions

The actions that you specify for your stateful rules help determine the order in which the Suricata stateful rules engine processes them. Network Firewall supports the Suricata rule actions pass, drop, and alert. By default, the engine processes rules in the order of pass action, drop action, then finally alert action. Within each action, you can set a priority to indicate processing order. For more information, see Evaluation order for stateful rule groups (p. 42).

Stateful rules can send alerts to the firewall's logs, if you have logging configured. To see the alerts, you must enable logging for the firewalls that use the rules. Logging incurs additional costs. For more information, see Logging network traffic from AWS Network Firewall (p. 83).

The options for stateful action settings vary by rule type.

Standard rules and Suricata compatible strings

You specify one of the following action options for both the rules that you provide in Suricata compatible strings and the rules that you specify using the standard, 5-tuple interface in Network Firewall. These options are a subset of the action options that are defined by Suricata. For more information, see Stateful rule groups in AWS Network Firewall (p. 41).

- **Pass** – Discontinue inspection of the matching packet and permit it to go to its intended destination. Rules with pass action are evaluated before rules with other action settings.
- **Drop** or **Alert**– Evaluate the packet against all rules with drop or alert action settings. If the firewall has alert logging configured, send a message to the firewall's alert logs for each matching rule. The first log entry for the packet will be for the first rule that matched the packet.

  After all rules have been evaluated, handle the packet according to the the action setting in the first rule that matched the packet. If the first rule has a drop action, block the packet. If it has an alert action, permit the packet to go to its intended destination.

Note
Matching a `drop` or `alert` rule for a packet doesn't necessarily mean the end of rule processing for that packet. The engine continues evaluating other rules for matches. For example, if there's a `drop` match that drops a packet, the packet can still go on to match an `alert` rule that generates alert logs. Matching an `alert` rule also doesn't imply a `pass`. The packet can go on to match a `drop` rule, and drop the packet after it's previously matched an `alert` rule.

For information about what you can do to manage the evaluation order of your stateful rules, see Evaluation order for stateful rule groups (p. 42).

Domain lists

The domain list rule group has one action setting at the rule group level. You specify one of the following options:

- **Allow** – Indicates that the domain name list is to be used as an allow list for all traffic that matches the specified protocols. For matching packets, discontinue inspection of the packet and permit it to pass to its intended destination. For non-matching packets, discontinue inspection of the packet, block it from going to its intended destination, and send a message to the firewall's alert logs if the firewall has alert logging configured.
- **Deny** – Indicates that the domain name list is to be used as a deny list for traffic that matches the specified protocols. For matching packets, discontinue inspection of the packet, block it from going to its intended destination, and send a message to the firewall's alert logs if the firewall has alert logging configured. For non-matching packets, take no action.

Managing your rule group in AWS Network Firewall

Follow the guidance in this section to manage your AWS Network Firewall rule groups.

How Network Firewall propagates your changes

When you make any changes to a firewall, including changes to any of the firewall's components, like rule groups and firewall policies, Network Firewall propagates the changes everywhere that the firewall is 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. For example, if you modify a rule group so that it drops an additional type of packet, for a firewall that uses the rule group, the new packet type might briefly be dropped by one firewall endpoint while still being allowed by another.

This temporary inconsistency can occur when you first create a firewall and when you make changes to an existing firewall. Generally, any inconsistencies of this type last only a few seconds.

When you update rules in a stateful rule group and the updates don't change the rule order, Network Firewall propagates the new rules without stopping and restarting the service. This minimizes service disruption for traffic flows that are already established. If the update does change from one rule order to another, the existing flows are still disrupted.

Changes to stateful rules are applied only to new traffic flows. Other firewall changes, including changes to stateless rules, are applied to all network packets.

Topics

Creating a stateless rule group

This section provides guidance for creating a stateless rule group through the console.

To create a stateless rule group

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Network Firewall rule groups**.
3. Choose **Create Network Firewall rule group**.
4. On the **Create Network Firewall rule group** page, for the **Rule group type**, choose **Stateless rule group**.
5. Enter a name and description for the rule group. You'll use these to identify the rule group when you manage it and use it.

   Note
   You can't change the name after you create the rule group.
6. For **Capacity**, set the maximum capacity you want to allow for the stateless rule group, up to the maximum of 30,000. You can't change this setting after you create the rule group. For information about how to calculate this, see Rule group capacity in AWS Network Firewall (p. 54). For information about the maximum setting, see AWS Network Firewall quotas (p. 101).
7. Review the rules that you want to add to the stateless rule group. Determine roughly what order you want Network Firewall to process them within the rule group. You need to provide unique, positive integer priority settings for your rules to indicate the processing order. Network Firewall processes from the lowest number up. We recommend using numbers with room in between, to allow for future insertions within the list of rules. For example, you might start with rule priorities numbered 100, 200, and so on.
8. Add each rule to the rule group as follows:

   a. For **Priority**, provide the priority to set the processing order of your rule.
   b. Choose the protocol and the source and destination settings for your rule.
   c. (Optional) For **TCP flags** provide the masks and flags that you want to inspect for. In **Masks**, indicate the flags that you want to inspect. In **Flags**, indicate which of the flags that you selected in **Masks** must be set. The other flags that you selected in **Masks** must be unset.
   d. For **Actions**, do the following:

      i. For **Action**, select the standard action that you want Network Firewall to take when a packet matches the rule settings.
      ii. (Optional) For **Custom actions**, add a new named custom action or select one that you've already created in the rule group. This option sends an Amazon CloudWatch metric dimension named `CustomAction` with a value that you specify.

      For additional information on these options, see Stateless actions (p. 55).
   e. Choose **Add rule**. Your rule is added to the **Rules** list for the rule group, ordered by priority.
9. Review the settings for the rule group, then choose **Create stateless rule group**.

Your new rule group is added to the list in the **Network Firewall rule groups** page.

To use your rule group in a firewall policy, follow the procedures at Managing your firewall policy (p. 35).

Creating a stateful rule group

This section provides guidance for creating a stateful rule group.

To create a stateful rule group

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Network Firewall rule groups**.
3. Choose **Create Network Firewall rule group**.
4. In the **Create Network Firewall rule group** page, for the **Rule group type**, choose **Stateful rule group**.

   For more information about stateful rule groups, see Stateful rule groups in AWS Network Firewall (p. 41).
5. Enter a name and description for the rule group. You'll use these to identify the rule group when you manage it and use it.

   Note
   You can't change the name after you create the rule group.
6. For **Capacity**, set the maximum capacity you want to allow for the stateful rule group, up to the maximum of 30,000. You can't change this setting after you create the rule group. For information about how to calculate this, see Rule group capacity in AWS Network Firewall (p. 54). For information about the maximum setting, see AWS Network Firewall quotas (p. 101).
7. Select the type of rule group that you want to add, from the **Stateful rule group options**. The rest of your rule group specifications depend on the option you choose.

   - (Option) **5-tuple** – Entry form for a basic Suricata rule.

     Note
     If you need to specify additional rule options, you can also use one of the APIs or AWS CloudFormation. For information, see StatefulRule in the *AWS Network Firewall API Reference* and AWS::NetworkFirewall::RuleGroup StatefulRule in the *AWS CloudFormation User Guide*.

     To choose the way that your stateful rules are ordered for evaluation, in the **Stateful rule order** section, choose a rule order:
     - Choose **Default** to have the stateful rules engine determine the evaluation order of your rules.
     - Choose **Strict** to provide your rules in the order that you want them to be evaluated.

     For each rule that you want in your rule group, specify the following information and then choose **Add rule**. Your added rules are listed in the **Rules** list.
     - Choose the protocol and source and destination settings for your rule.
     - For **Traffic direction**, choose whether to apply the rule to any direction or only for traffic that flows forward, from the specified source to the specified destination.
     - For **Action**, select the action that you want Network Firewall to take when a packet matches the rule settings. For information on these options, see Stateful actions (p. 55).

     For information about these rules, see Standard stateful rule groups in AWS Network Firewall (p. 48).

   - (Option) **Domain list** – Specify the following information.
     - For **Domain name source**, enter the domain names that you want to inspect for, one name specification per line. Valid domain name specifications are the following:
       - Explicit names. For example, `abc.example.com` matches only the domain `abc.example.com`.
       - Names that use a domain wildcard, which you indicate with an initial '`.`'. For example, `.example.com` matches `example.com` and matches all subdomains of `example.com`, such as `abc.example.com` and `www.example.com`.

- For **Protocols**, choose the protocols you want to inspect for.
- For **Action**, select the list type that you are creating, either **Allow** or **Deny**. For information on these options, see Stateful actions (p. 55).

For information about stateful domain name rules, see Stateful domain list rule groups in AWS Network Firewall (p. 45).

- (Option) **Suricata compatible IPS rules**

  To choose the way that your stateful rules are ordered for evaluation, in the **Stateful rule order** section, choose a rule order:
  - Choose **Default** to have the stateful rules engine determine the evaluation order of your rules.
  - Choose **Strict** to provide your rules in the order that you want them to be evaluated.

  Paste your rules into the text box.

8. Review the settings that you've provided for the rule group, then choose **Create stateful rule group**.

Your new rule group is added to the list in the **Network Firewall rule groups** page.

To use your rule group in a firewall policy, follow the procedures at Managing your firewall policy (p. 35).

Updating a rule group

To change your rule group settings, use the following procedure.

To update a rule group

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Network Firewall rule groups**.
3. In the **Network Firewall rule groups** page, choose the name of the rule group that you want to update. The rule group's details page appears.
4. In your rule group's details page, in the area that you want to change, choose **Edit**. Follow the prompts to make your updates. The interface varies according to the rule group type. When you're done editing an area, choose **Save** to save your changes in the rule group.

How Network Firewall propagates your changes

When you make any changes to a firewall, including changes to any of the firewall's components, like rule groups and firewall policies, Network Firewall propagates the changes everywhere that the firewall is 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. For example, if you modify a rule group so that it drops an additional type of packet, for a firewall that uses the rule group, the new packet type might briefly be dropped by one firewall endpoint while still being allowed by another.

This temporary inconsistency can occur when you first create a firewall and when you make changes to an existing firewall. Generally, any inconsistencies of this type last only a few seconds.

When you update rules in a stateful rule group and the updates don't change the rule order, Network Firewall propagates the new rules without stopping and restarting the service. This minimizes service disruption for traffic flows that are already established. If the update does change from one rule order to another, the existing flows are still disrupted.

Changes to stateful rules are applied only to new traffic flows. Other firewall changes, including changes to stateless rules, are applied to all network packets.

Deleting a rule group

To delete a rule group, use the guidance in this section.

Deleting a rule group or firewall policy

When you delete a rule group or a firewall policy, AWS Network Firewall checks to see if it's currently being referenced. A rule group can be referenced by a firewall policy, and a firewall policy can be referenced by a firewall. If Network Firewall determines that the resource is being referenced, it warns you. Network Firewall is almost always able to determine whether a resource is being referenced. However, in rare cases, it might not be able to do so. If you need to be sure that the resource that you want to delete isn't in use, check all of your firewalls or firewall policies before deleting it. Note that policies that have associations can't be deleted.

To delete a rule group

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Network Firewall rule groups**.
3. In the **Network Firewall rule groups** page, select the name of the rule group that you want to delete, and then choose **Delete**.

How Network Firewall propagates your changes

When you make any changes to a firewall, including changes to any of the firewall's components, like rule groups and firewall policies, Network Firewall propagates the changes everywhere that the firewall is 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. For example, if you modify a rule group so that it drops an additional type of packet, for a firewall that uses the rule group, the new packet type might briefly be dropped by one firewall endpoint while still being allowed by another.

This temporary inconsistency can occur when you first create a firewall and when you make changes to an existing firewall. Generally, any inconsistencies of this type last only a few seconds.

Changes to stateful rules are applied only to new traffic flows. Other firewall changes, including changes to stateless rules, are applied to all network packets.

Sharing firewall policies and rule groups

The owner of a firewall policy or rule group can share a resource with:

- Specific AWS accounts inside or outside of its organization in AWS Organizations
- An organizational unit inside its organization in AWS Organizations
- Its entire organization in AWS Organizations

Contents

Prerequisites for sharing firewall policies and rule groups

- To share a firewall policy or rule group, you must own it in your AWS account. You cannot share a firewall policy or rule group that has been shared with you.
- To share a firewall policy or rule group with your organization or an organizational unit in AWS Organizations, you must enable sharing with AWS Organizations. For more information, see Enable Sharing with AWS Organizations in the *AWS RAM User Guide*.

Related services

Firewall policy and rule group sharing integrates with AWS Resource Access Manager (AWS RAM). AWS RAM is a service that enables you to share your AWS resources with any AWS account or through AWS Organizations. With AWS RAM, you share resources that you own by creating a *resource share*. A resource share specifies the resources to share, and the consumers with whom to share them. Consumers can be individual AWS accounts, organizational units, or an entire organization in AWS Organizations.

For more information about AWS RAM, see the *AWS RAM User Guide*.

Sharing across Availability Zones

To ensure that resources are distributed across the Availability Zones for a Region, we independently map Availability Zones to names for each account. This could lead to Availability Zone naming differences across accounts. For example, the Availability Zone `us-east-1a` for your AWS account might not have the same location as `us-east-1a` for another AWS account.

To identify the location of your firewall policy or rule group relative to your accounts, you must use the *Availability Zone ID* (AZ ID). The AZ ID is a unique and consistent identifier for an Availability Zone across all AWS accounts. For example, `use1-az1` is an AZ ID for the `us-east-1` Region and it is the same location in every AWS account.

To view the AZ IDs for the Availability Zones in your account

1. Open the AWS RAM console at https://console.aws.amazon.com/ram.
2. The AZ IDs for the current Region are displayed in the **Your AZ ID** panel on the right-hand side of the screen.

Sharing a firewall policy or rule group

To share a firewall policy or rule group, you must add it to a resource share. A resource share is an AWS RAM resource that lets you share your resources across AWS accounts. A resource share specifies the resources to share, and the consumers with whom they are shared. When you share a firewall policy or rule group using AWS Network Firewall, you add it to an existing resource share. To add the firewall policy or rule group to a new resource share, you must first create the resource share using the AWS RAM console.

If you are part of an organization in AWS Organizations and sharing within your organization is enabled, consumers in your organization are automatically granted access to the shared firewall policies and rule groups. Otherwise, consumers receive an invitation to join the resource share and are granted access to the shared firewall policies and rule groups after accepting the invitation.

You can share a firewall policy or rule group that you own using the AWS RAM console, the AWS Network Firewall API, or the AWS CLI.

To share a firewall policy or rule group that you own using the AWS RAM console

See Creating a Resource Share in the *AWS RAM User Guide*.

To share a firewall policy or rule group that you own using the AWS CLI

Use the create-resource-share command.

To share a firewall policy or rule group that you own using the Network Firewall API

Use the `PutResourcePolicy` action. For information about how to use this, see PutResourcePolicy in the *AWS Network Firewall API Reference*.

Unsharing a shared firewall policy or rule group

To unshare a shared firewall policy or rule group that you own, you must remove it from the resource share. You can do this using the AWS RAM console or the AWS CLI.

To unshare a shared firewall policy or rule group that you own using the AWS RAM console

See Updating a Resource Share in the *AWS RAM User Guide*.

To unshare a shared firewall policy or rule group that you own using the AWS CLI

Use the disassociate-resource-share command.

Security in AWS Network Firewall

This section describes AWS Cloud security for Network Firewall. 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 Network Firewall, 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 Network Firewall. The following topics show you generally how to configure Network Firewall to meet your security and compliance objectives for using an AWS service. You also learn how to use other AWS services that help you to monitor and secure your Network Firewall resources.

Use this general guidance in addition to the guidance for using the AWS Network Firewall service itself. Network Firewall is intended to improve the security of communication into and out of your Amazon Virtual Private Cloud VPCs, and this entire developer guide provides guidance for using Network Firewall.

Topics

Data protection in Network Firewall

The AWS shared responsibility model applies to data protection in AWS Network Firewall. 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 Network Firewall 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.

Network Firewall entities—such as firewalls, firewall policies, and rule groups—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 for AWS Network Firewall

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use Network Firewall resources. IAM is an AWS service that you can use with no additional charge.

Topics

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in Network Firewall.

Service user – If you use the Network Firewall service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more Network Firewall features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in Network Firewall, see Troubleshooting AWS Network Firewall identity and access (p. 79).

Service administrator – If you're in charge of Network Firewall resources at your company, you probably have full access to Network Firewall. It's your job to determine which Network Firewall features and resources your employees should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the

basic concepts of IAM. To learn more about how your company can use IAM with Network Firewall, see How AWS Network Firewall works with IAM (p. 68).

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to Network Firewall. To view example Network Firewall identity-based policies that you can use in IAM, see AWS Network Firewall identity-based policy examples (p. 73).

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. For more information about signing in using the AWS Management Console, see Signing in to the AWS Management Console as an IAM user or root user in the *IAM User Guide*.

You must be *authenticated* (signed in to AWS) as the AWS account root user, an IAM user, or by assuming an IAM role. You can also use your company's single sign-on authentication or even sign in using Google or Facebook. In these cases, your administrator previously set up identity federation using IAM roles. When you access AWS using credentials from another company, you are assuming a role indirectly.

To sign in directly to the AWS Management Console, use your password with your root user email address or your IAM user name. You can access AWS programmatically using your root user or IAM users access keys. AWS provides SDK and command line tools to cryptographically sign your request using your credentials. If you don't use AWS tools, you must sign the request yourself. Do this using *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*.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Using multi-factor authentication (MFA) in AWS in the *IAM User Guide*.

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 users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. An IAM user can have long-term credentials such as a user name and password or a set of access keys. To learn how to generate access keys, see Managing access keys for IAM users in the *IAM User Guide*. When you generate access keys for an IAM user, make sure you view and securely save the key pair. You cannot recover the secret access key in the future. Instead, you must generate a new access key pair.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see When to create an IAM user (instead of a role) in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by switching roles. You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see Using IAM roles in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.
- **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*.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
  - **Principal permissions** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. Policies grant permissions to a principal. When you use some services, you might perform an action that then triggers another action in a different service. In this case, you must have permissions to perform both actions. To see whether an action requires additional dependent actions in a policy, see Actions, Resources, and Condition Keys for AWS Network Firewall in the *Service Authorization Reference*.
  - **Service role** – 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*.
  - **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **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*.

To learn whether to use IAM roles or IAM users, see When to create an IAM role (instead of a user) in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to IAM identities or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions.

You can sign in as the root user or an IAM user, or you can assume an IAM role. When you then make a request, AWS evaluates the related identity-based or resource-based policies. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

Every IAM entity (user or role) starts with no permissions. In other words, by default, users can do nothing, not even change their own password. To give a user permission to do something, an administrator must attach a permissions policy to a user. Or the administrator can add the user to a group that has the intended permissions. When an administrator gives permissions to a group, all users in that group are granted those permissions.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Creating IAM policies in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choosing between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see Access control list (ACL) overview in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see How SCPs work in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How AWS Network Firewall works with IAM

To use IAM to manage access to Network Firewall, understand what IAM features are available to use with Network Firewall. To get a high-level view of how Network Firewall and other AWS services work with IAM, see AWS services that work with IAM in the *IAM User Guide*.

Topics

Network Firewall identity-based policies

With IAM identity-based JSON policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied.

Network Firewall supports specific actions, resources, and condition keys. To learn about all of the elements that you use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Actions

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as *permission-only actions* that don't have a matching API operation. There are also

some operations that require multiple actions in a policy. These additional actions are called *dependent actions*.

Include actions in a policy to grant permissions to perform the associated operation.

Policy actions in Network Firewall use the following prefix before the action: `network-firewall:`. For example, to grant someone permission to create a Network Firewall firewall using the `CreateFirewall` API operation, you include the `network-firewall:CreateFirewall` action in their policy. Policy statements must include either an `Action` or `NotAction` element. Network Firewall defines its own set of actions that describe tasks that you can perform with this service.

To specify multiple actions in a single statement, separate them with commas as follows:

```
"Action": [
      "network-firewall:action1",
      "network-firewall:action2"
```

You can specify multiple actions using wildcards (*). For example, to specify all actions that begin with the word `Describe`, include the following action:

```
"Action": "network-firewall:Describe*"
```

As a best practice, avoid using wildcards. Grant the least privilege that supports the operations that your users require.

The following example policy shows the use of various action specifications. Some are for specific, named resources, and some are for all resources. This policy provides permission to retrieve the lists of all firewalls and firewall policies, to retrieve a specific named firewall and associate a firewall policy with it, and to retrieve two named Network Firewall firewall policies. This policy doesn't provide permission to do things like create a new firewall or update the configuration of a firewall policy.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":["network-firewall:ListFirewalls", "network-
firewall:ListFirewallPolicies"],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "network-firewall:AssociateFirewallPolicy",
            "network-firewall:DescribeFirewall",
         ],
         "Resource":"arn:aws:network-firewall:::firewall/my-firewall/*"
      },
      {
         "Effect":"Allow",
         "Action":["network-firewall:DescribeFirewallPolicy"],
         "Resource": [
            "arn:aws:network-firewall:::policy/my-firewall-policy",
            "arn:aws:network-firewall:::policy/second-policy"
         ]
      }
   ]
}
```

The following example policy provides permission for the read actions that are available in Network Firewall.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "network-firewall:List*",
            "network-firewall:Describe*"
         ],
         "Resource":"*"
      }
   ]
}
```

To see a list of all Network Firewall actions, see Actions, Resources, and Condition Keys for AWS Network Firewall in the *Service Authorization Reference*.

Resources

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. As a best practice, specify a resource using its Amazon Resource Name (ARN). You can do this for actions that support a specific resource type, known as *resource-level permissions*.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

In AWS Network Firewall, the resources include *firewalls*, *firewall policies*, and stateless and stateful *rule groups*. These resources have unique Amazon Resource Names (ARNs) associated with them, as shown in the following table.

Name in AWS Network Firewall console	Name in AWS Network Firewall SDK/ CLI	ARN format	
Firewall	`Firewall`	`arn:aws:network-firewall:`*`region`*`:`*`account`*`:firewall/`*`ID`	
Firewall policy	`FirewallPolicy`	`arn:aws:network-firewall:`*`region`*`:`*`account`*`:firewall-policy/`*`ID`	
Stateful rule group	`RuleGroup` with `Type="STATEFUL"`	`arn:aws:network-firewall:`*`region`*`:`*`account`*`:stateful-rulegroup/`*`ID`	
Stateless rule group	`RuleGroup` with `Type="STATELESS"`	`arn:aws:network-firewall:`*`region`*`:`*`account`*`:stateless-rulegroup/`*`ID`	

To allow or deny access to a subset of Network Firewall resources, include the ARN specification in the `resource` element of your policy. For example, the following ARN specifies all firewall resources for the account `111122223333` in Region `us-east-1`:

```
arn:aws:network-firewall:us-east-1:111122223333:firewall/*
```

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

AWS Network Firewall provides a set of operations to work with Network Firewall resources. For a list of available operations, see Actions in the *AWS Network Firewall API Reference*.

For more information about the format of ARNs, see Amazon Resource Names (ARNs) and AWS service namespaces.

Some Network Firewall actions, such as those for creating resources, can't be performed on a resource that you can specify ahead of time. For those cases, you must use the wildcard (*).

```
"Resource": "*"
```

Some AWS Network Firewall API actions work on multiple resources. For example, `CreateFirewall` modifies the subnets that you specify for the firewall endpoints, so a Network Firewall user must have permissions to update your VPC subnets. To specify multiple resources in a single statement, separate the ARNs with commas.

```
"Resource": [
      "resource1",
      "resource2"
```

To see a list of Network Firewall resource types and their ARNs and to learn with which actions you can specify the ARN of each resource, see Actions, Resources, and Condition Keys for AWS Network Firewall in the *Service Authorization Reference*.

Condition keys

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element (or `Condition` *block*) lets you specify conditions in which a statement is in effect. The `Condition` element is optional. You can create conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple `Condition` elements in a statement, or multiple keys in a single `Condition` element, AWS evaluates them using a logical `AND` operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical `OR` operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see IAM policy elements: variables and tags in the *IAM User Guide*.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

Network Firewall doesn't define any condition keys of its own, but it supports using some global condition keys, including the ones used for tagging, such as `aws:TagKeys`. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

For more information about Network Firewall actions, resources, and condition keys, see Actions, Resources, and Condition Keys for AWS Network Firewall in the *Service Authorization Reference*.

Examples

To view examples of Network Firewall identity-based policies, see AWS Network Firewall identity-based policy examples (p. 73).

Network Firewall resource-based policies

Resource-based policies are JSON policy documents that specify what actions a specified principal can perform on a resource and under what conditions. For example, Amazon S3 supports resource-based permissions policies for Amazon S3 buckets. Resource-based policies let you grant usage permission to other accounts on a per-resource basis. You can also use a resource-based policy to allow an AWS service to access your Amazon S3 buckets.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. Adding a cross-account principal to a resource-based policy is only half of the work you need to do to establish the trust relationship. When the principal and the resource are in different AWS accounts, you must also grant the principal entity permission to access the resource. Grant permission by attaching an identity-based policy to the entity. However, if a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required. For more information, see How IAM Roles Differ from Resource-based Policies in the *IAM User Guide*.

AWS Network Firewall supports resource-based policies for resource sharing, for the resources firewall policy and rule group. These policies define which principal entities (accounts, users, roles, and federated users) can perform actions on a resource. To learn more, see Sharing firewall policies and rule groups (p. 61).

Authorization based on Network Firewall tags

You can attach tags to Network Firewall resources or pass tags in a request to Network Firewall. To control access based on tags, you provide tag information in the condition element of a policy using the `network-firewall:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, or `aws:TagKeys` condition keys.

Network Firewall IAM roles

An IAM role is an entity within your AWS account that has specific permissions.

Using temporary credentials with Network Firewall

You can use temporary credentials to sign in with federation, assume an IAM role, or to assume a cross-account role. You obtain temporary security credentials by calling AWS STS API operations such as AssumeRole or GetFederationToken.

Network Firewall supports using temporary credentials.

Service-linked roles

Service-linked roles allow AWS services to access resources in other services to complete an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view but not edit the permissions for service-linked roles.

Network Firewall supports service-linked roles.

Service roles

This feature allows a service to assume a service role on your behalf. This role allows the service to access resources in other services to complete an action on your behalf. Service roles appear in your IAM account and are owned by the account. This means that an IAM administrator can change the permissions for this role. However, doing so might break the functionality of the service.

Network Firewall supports service roles for firewall logging activities. When you configure your logging destination, depending on the destination type and whether and how you use encryption, you might need to define one or more roles for your logging activities. For information, see Logging network traffic from AWS Network Firewall (p. 83).

AWS Network Firewall identity-based policy examples

By default, IAM users and roles don't have permission to create or modify Network Firewall resources. They also can't perform tasks using the AWS Management Console, AWS CLI, or AWS API. An IAM administrator must create IAM policies that grant users and roles permission to perform specific API operations on the specified resources they need. The administrator must then attach those policies to the IAM users or groups that require those permissions.

To learn how to create an IAM identity-based policy using these example JSON policy documents, see Creating policies on the JSON tab in the *IAM User Guide*.

Topics

Policy best practices

Identity-based policies are very powerful. They determine whether someone can create, access, or delete Network Firewall resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started using AWS managed policies** – To start using Network Firewall quickly, use AWS managed policies to give your employees the permissions they need. These policies are already available in your account and are maintained and updated by AWS. For more information, see Get started using permissions with AWS managed policies in the *IAM User Guide*.
- **Grant least privilege** – When you create custom policies, grant only the permissions required to perform a task. Start with a minimum set of permissions and grant additional permissions as necessary. Doing so is more secure than starting with permissions that are too lenient and then trying to tighten them later. For more information, see Grant least privilege in the *IAM User Guide*.
- **Enable MFA for sensitive operations** – For extra security, require IAM users to use multi-factor authentication (MFA) to access sensitive resources or API operations. For more information, see Using multi-factor authentication (MFA) in AWS in the *IAM User Guide*.
- **Use policy conditions for extra security** – To the extent that it's practical, define the conditions under which your identity-based policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also write conditions

to allow requests only within a specified date or time range, or to require the use of SSL or MFA. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.

Using the Network Firewall console

The AWS Network Firewall console resides inside the Amazon Virtual Private Cloud (Amazon VPC) console, so you must have the permissions documented at Viewing the Amazon VPC console in the Amazon Virtual Private Cloud User Guide. These permissions allow you to list and view details about the Network Firewall resources in your AWS account.

If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (IAM users or roles) that use that policy.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that you're trying to perform.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Viewing Network Firewall resources based on tags

You can use conditions in your identity-based policy to control access to Network Firewall resources based on tags. This example shows how you might create a policy that allows viewing a firewall.

However, permission is granted only if the firewall tag `Owner` has the value of that user's user name. This policy also grants the permissions necessary to complete this action on the console.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListFirewallsInConsole",
            "Effect": "Allow",
            "Action": "network-firewall:ListFirewalls",
            "Resource": "*"
        },
        {
            "Sid": "ViewFirewallIfOwner",
            "Effect": "Allow",
            "Action": "network-firewall:DescribeFirewall",
            "Resource": "arn:aws:network-firewall:*:*:firewall/*",
            "Condition": {
                "StringEquals": {"network-firewall:ResourceTag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

You can attach this policy to the IAM users in your account. If a user named `richard-roe` attempts to view a Network Firewall firewall, the firewall must be tagged `Owner=richard-roe` or `owner=richard-roe`. Otherwise Network Firewall denies the access. The condition tag key `Owner` matches both `Owner` and `owner` because condition key names are not case-sensitive. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.

Using service-linked roles for Network Firewall

AWS Network Firewall 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 Network Firewall. Service-linked roles are predefined by Network Firewall and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up Network Firewall easier because you don’t have to manually add the necessary permissions. Network Firewall defines the permissions of its service-linked roles, and unless defined otherwise, only Network Firewall 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 its related resources. This protects your Network Firewall 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 Network Firewall

Network Firewall uses the service-linked role named **`AWSServiceRoleForNetworkFirewall`** – An access policy that allows AWS Network Firewall to manage Network Firewall related resources on behalf of your AWS account. Network Firewall uses its service-linked-role to create, describe, and delete VPC endpoints in support of your firewall management activities. Network Firewall is the only service that uses this service-linked role, and Network Firewall doesn't use any other service's service-linked role. This service-linked role is used in the Network Firewall managed policy

`AWSNetworkFirewallServiceRolePolicy`. For more information, see AWS managed policies for AWS Network Firewall (p. 78).

The `AWSServiceRoleForNetworkFirewall` service-linked role trusts the `network-firewall.amazonaws.com` service principal to assume the role. The following is the JSON trust policy:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "network-firewall.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

The role permissions policy allows Network Firewall to perform actions on Amazon EC2 VPC resources for firewall management. The actions include periodic checks on the VPC CIDR blocks and management of firewall endpoints in the VPC. The following is the JSON role permissions policy:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:CreateVpcEndpoint",
                "ec2:DescribeVpcEndpoints"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*:*:vpc-endpoint/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateVpcEndpoint",
                    "aws:RequestTag/AWSNetworkFirewallManaged": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteVpcEndpoints"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/AWSNetworkFirewallManaged": "true"
                }
            }
        }
```

```
    ]
}
```

When you enable logging for a firewall, Network Firewall uses a log delivery service, which might create a service-linked role in your account named `AWSServiceRoleForLogDelivery` to deliver logs.

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 Network Firewall

You don't need to manually create a service-linked role for AWS Network Firewall. When you create a AWS Network Firewall firewall in the AWS Management Console, the AWS CLI, or the AWS API, if your account doesn't have the Network Firewall service-linked role yet, Network Firewall creates the role for you. If you manage your firewall resources through AWS Firewall Manager, Firewall Manager creates the service-linked role for accounts that are within scope of the Firewall Manager policy. If you want to, you can create the role through the IAM console. If you delete the service-linked role, the next time you create an Network Firewall resource, Network Firewall creates one for you again.

Editing a service-linked role for Network Firewall

Network Firewall doesn't allow you to edit the `AWSServiceRoleForNetworkFirewall` service-linked role. After you create a service-linked role, you can't change the name of the role because various entities might reference it. 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 Network Firewall

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. You must clean up the resources that require your service-linked role before you can manually delete it. You can delete the service-linked role used by Network Firewall if you no longer want to use the service. To delete the role, you must delete all firewalls, firewall policies, stateful rule groups, and stateless rule groups, in all Regions where you have them defined.

> **Note**
> If the Network Firewall 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 all Network Firewall resources

1. On the Amazon VPC console, update all route tables that send traffic through your firewall endpoints, to remove the endpoints from the traffic flow. For information about managing route tables for your VPC, see Route tables in the *Amazon Virtual Private Cloud User Guide*.
2. On the Network Firewall console, remove your firewalls, firewall policies, stateful rules groups, and stateless rule groups. For information, see Deleting a firewall (p. 32), Deleting a firewall policy (p. 38), and Deleting a rule group (p. 60).

This removes all resources that Network Firewall used the service-linked role for.

To manually delete the service-linked role using IAM

Use the IAM console, the IAM CLI, or the IAM API to delete the `AWSServiceRoleForNetworkFirewall` service-linked role. For more information, see Deleting a Service-Linked Role in the *IAM User Guide*.

Supported Regions for Network Firewall service-linked roles

Network Firewall supports using service-linked roles in all of the Regions where the service is available. For a Region list, see AWS Regions and Endpoints.

AWS managed policies for AWS Network Firewall

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*.

Network Firewall updates to AWS managed policies

View details about updates to AWS managed policies for Network Firewall since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the Network Firewall Document history page.

Change	Description	Date
`AWSNetworkFirewallServiceRolePolicy` – Update to the existing policy	AWS Network Firewall expanded availability of the policy to the AWS GovCloud (US) Regions, AWS GovCloud (US-East) and AWS GovCloud (US-West). `AWSNetworkFirewallServiceRolePolicy` is an access policy that allows Network Firewall to manage Network Firewall related resources on behalf of your AWS account. Network Firewall uses this policy to create, describe, and delete VPC endpoints in support of your firewall management activities. For policy details, see AWSNetworkFirewallServiceRolePolicy. This policy uses the service-linked role `AWSServiceRoleForNetworkFirewall`. For more information, see Using service-linked roles for Network Firewall (p. 75).	June 24, 2021

Change	Description	Date
Network Firewall started tracking changes	Network Firewall started tracking changes for its AWS managed policies.	June 24, 2021

Troubleshooting AWS Network Firewall identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with Network Firewall and IAM.

Topics

I am not authorized to perform an action in Network Firewall

If the AWS Management Console tells you that you're not authorized to perform an action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a firewall but does not have `network-firewall:DescribeFirewall` permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform: network-
firewall:DescribeFirewall on resource: exampleFirewall
```

In this case, Mateo asks his administrator to update his policies to allow him to access the firewall resource using the `network-firewall:DescribeFirewall` action.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password. Ask that person to update your policies to allow you to pass a role to Network Firewall.

Some AWS services allow you to pass an existing role to that service, instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in Network Firewall. However, the action requires the service to have permissions granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform: iam:PassRole
```

In this case, Mary asks her administrator to update her policies to allow her to perform the `iam:PassRole` action.

I want to view my access keys

After you create your IAM user access keys, you can view your access key ID at any time. However, you can't view your secret access key again. If you lose your secret key, you must create a new access key pair.

Access keys consist of two parts: an access key ID (for example, `AKIAIOSFODNN7EXAMPLE`) and a secret access key (for example, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Like a user name and password, you must use both the access key ID and secret access key together to authenticate your requests. Manage your access keys as securely as you do your user name and password.

> **Important**
> Do not provide your access keys to a third party, even to help find your canonical user ID. By doing this, you might give someone permanent access to your account.

When you create an access key pair, you are prompted to save the access key ID and secret access key in a secure location. The secret access key is available only at the time you create it. If you lose your secret access key, you must add new access keys to your IAM user. You can have a maximum of two access keys. If you already have two, you must delete one key pair before creating a new one. To view instructions, see Managing access keys in the *IAM User Guide*.

I'm an administrator and want to allow others to access Network Firewall

To allow others to access Network Firewall, you must create an IAM entity (user or role) for the person or application that needs access. They will use the credentials for that entity to access AWS. You must then attach a policy to the entity that grants them the correct permissions in Network Firewall.

To get started right away, see Creating your first IAM delegated user and group in the *IAM User Guide*.

I want to allow people outside of my AWS account to access my Network Firewall resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether Network Firewall supports these features, see How AWS Network Firewall works with IAM (p. 68).
- To learn how to provide access to your resources across AWS accounts that you own, see Providing access to an IAM user in another AWS account that you own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing access to AWS accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the *IAM User Guide*.

AWS logging and monitoring tools

This section provides an overview of the tools available for logging and monitoring in AWS Network Firewall for standard AWS security purposes. For more information about logging and monitoring in Network Firewall see Logging and monitoring in AWS Network Firewall (p. 83).

Monitoring is an important part of maintaining the reliability, availability, and performance of Network Firewall 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 Network Firewall resources and responding to potential incidents:

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 AWS Network Firewall metrics in Amazon CloudWatch (p. 97).

AWS CloudTrail Logs

CloudTrail provides a record of actions taken by a user, role, or an AWS service in Network Firewall. Using the information collected by CloudTrail, you can determine the request that was made to Network Firewall, 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 calls to the AWS Network Firewall API with AWS CloudTrail (p. 92).

AWS Trusted Advisor

Trusted Advisor draws upon best practices learned from serving hundreds of thousands of AWS customers. Trusted Advisor inspects your AWS environment and then makes recommendations when opportunities exist to save money, improve system availability and performance, or help close security gaps. All AWS customers have access to five Trusted Advisor checks. Customers with a Business or Enterprise support plan can view all Trusted Advisor checks. For more information, see AWS Trusted Advisor.

Compliance validation for Network Firewall

Third-party auditors assess the security and compliance of Network Firewall, as part of a number of AWS compliance programs. For general information about compliance programs, see AWS Compliance Programs. You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

You can see the current compliance status of Network Firewall and other AWS services at AWS Services in Scope by Compliance Program.

When you use Network Firewall, your compliance responsibility is determined by the sensitivity of your data, your organization's compliance objectives, and applicable laws and regulations. If your use of Network Firewall is subject to compliance with standards like HIPAA, PCI, or FedRAMP, AWS provides the following resources to help you:

- 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 Network Firewall

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 Network Firewall

As a managed service, AWS Network Firewall 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 Network Firewall 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.

Logging and monitoring in AWS Network Firewall

Logging and monitoring help you to maintain the reliability, availability, and performance of AWS Network Firewall. You can monitor how the service is being used and you can monitor network traffic and traffic filtering done by the stateful rule groups in your Network Firewall firewalls.

AWS provides a number of tools that you can use to monitor Network Firewall. 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.

You can use the following automated monitoring tools with Network Firewall:

- *Amazon CloudWatch* provides metrics for the AWS resources and the applications that you run on AWS. Monitoring and alarms are real time. You can collect and track metrics, create customized dashboards, and set alarms that notify you or take actions when a specified metric reaches a threshold that you specify. For example, you can have CloudWatch track CPU usage or other metrics of your Amazon EC2 instances and automatically launch new instances when needed. For more information, see the Amazon CloudWatch User Guide.
- *Amazon CloudWatch Logs* provides logging for sources such as Amazon EC2 instances and CloudTrail. CloudWatch Logs can monitor information in the log files and notify you when certain thresholds are met. You can also archive your log data in highly durable storage. For more information, see the Amazon CloudWatch Logs User Guide.
- *AWS CloudTrail* captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon S3 bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. For more information, see the AWS CloudTrail User Guide.
- *AWS Config* lets you view the configuration of your AWS resources in your AWS account. The available information includes 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. For more information, see the AWS Config Developer Guide.

Topics

Logging network traffic from AWS Network Firewall

You can configure AWS Network Firewall logging for your firewall's stateful engine. Logging gives you detailed information about network traffic, including the time that the stateful engine received a packet, detailed information about the packet, and any stateful rule action taken against the packet. The logs are published to the log destination that you've configured, where you can retrieve and view them.

Note
Firewall logging is only available for traffic that you forward to the stateful rules engine. You forward traffic to the stateful engine through stateless rule actions and stateless default actions in the firewall policy. For information about these actions settings, see Stateless default actions in your firewall policy (p. 35) and Rule actions in AWS Network Firewall (p. 55).
Metrics provide some higher-level information for both stateless and stateful engine types. For more information, see AWS Network Firewall metrics in Amazon CloudWatch (p. 97).

You can record flow logs and alert logs from your Network Firewall stateful engine.

- Flow logs are standard network traffic flow logs. Each flow log record captures the network flow for a specific 5-tuple.
- Alert logs report traffic that matches your stateful rules that have an action that sends an alert. A stateful rule sends alerts for the rule actions `DROP` and `ALERT`.

You can use the same or different logging destination for each log type. You enable logging for a firewall after you create it. For information about how to do this, see Updating a firewall's logging configuration (p. 92).

Contents of a firewall log

The Network Firewall logs contain the following information:

- **firewall_name** – The name of the firewall that's associated with the log entry.
- **availability_zone** – The Availability Zone of the firewall endpoint that generated the log entry.
- **event_timestamp** – The time that the log was created, written in epoch seconds at Coordinated Universal Time (UTC).
- **event** – Detailed information about the event. This information includes the event timestamp converted to human readable format, event type, network packet details, and, if applicable, details about the stateful rule that the packet matched against. The event is controlled by Suricata, the open source intrusion prevention system (IPS) that the stateful rules engine runs on. Suricata writes the event information in the Suricata EVE JSON output format.
  - The engine writes flow log events using the EVE output type `netflow`. The log type `netflow` logs uni-directional flows, so each event represents traffic going in a single direction.
  - The engine writes the alert log events using the EVE output type `alert`.

  For detailed information about these Suricata events, see EVE JSON Output in the Suricata User Guide.

The following shows an example alert log entry for Network Firewall:

```
{"firewall_name":"test-firewall","availability_zone":"us-
east-1b","event_timestamp":"1602627001","event":
{"timestamp":"2020-10-13T22:10:01.006481+0000","flow_id":1582438383425873,"event_type":"alert","src_ip"
{"action":"allowed","signature_id":5,"rev":0,"signature":"test_tcp","category":"","severity":1}}}
```

Firewall log delivery

A log file or log stream generally contains information about the requests that your firewall received during a given time period. The timing of Network Firewall log delivery varies by location type, averaging 3-6 minutes for Amazon CloudWatch Logs and Amazon Kinesis Data Firehose and 8-12 minutes for Amazon Simple Storage Service buckets. In some cases, logs may take longer than these averages. When

log entries are delayed, Network Firewall saves them and then logs them according to the date and time of the period in which the requests occurred, not the date and time when the logs are delivered.

Note
If your firewall doesn't filter traffic for a period of time, you don't receive logs for that period.

When creating a log file or stream, Network Firewall consolidates information for your firewall from all the endpoints that received traffic during the time period that the log covers.

Permissions to configure firewall logging

You must have the following permissions to make any changes to your firewall logging configuration. These settings are included in the permissions requirements for each logging configuration type, under AWS Network Firewall logging destinations (p. 85).

```
{
    "Action": [
        "logs:CreateLogDelivery",
        "logs:GetLogDelivery",
        "logs:UpdateLogDelivery",
        "logs:DeleteLogDelivery",
        "logs:ListLogDeliveries"
    ],
    "Resource": [
        "*"
    ],
    "Effect": "Allow",
    "Sid": "FirewallLogging"
}
```

The permissions required for logging configuration are in addition to the standard permissions required to use the Network Firewall API. For information about the standard permissions that are required to use Network Firewall, see Managing access using policies (p. 66).

Pricing for firewall logging

You are charged for logging, on top of the basic charges for using Network Firewall. Your costs can vary depending on factors such as the destination type that you choose and the amount of data that you log. For example, flow logging sends logs for all of the network traffic that reaches your firewall's stateful rules, but alert logging sends logs only for network traffic that your stateful rules drop or explicitly alert on. For information on Network Firewall pricing, see Pricing for AWS Network Firewall (p. 3).

Topics

AWS Network Firewall logging destinations

This section describes the logging destinations that you can choose from for your Network Firewall logs. Each section provides guidance for configuring logging for the destination type and information about any behavior that's specific to the destination type. After you've configured your logging destination, you can provide its specifications to the firewall logging configuration to start logging to it.

Topics

Amazon Simple Storage Service

To send your firewall logs to Amazon S3, you need to set up an Amazon S3 bucket for the logs. In your bucket configuration for the firewall, you can optionally include a prefix, to immediately follow the bucket name. When you enable logging in Network Firewall, you provide the bucket name and, if you are using one, the prefix. For information about creating your logging bucket, see Create a Bucket in the *Amazon Simple Storage Service User Guide*.

> **Note**
> Network Firewall supports encryption with Amazon S3 buckets for key type Amazon S3 key (SSE-S3) and for AWS Key Management Service (SSE-KMS) AWS KMS keys. Network Firewall doesn't support encryption for AWS Key Management Service keys that are managed by AWS.

Your alert and flow logs collect log records, consolidate them into log files, and then publish the log files to the Amazon S3 bucket at 5-minute intervals. Each log file contains log records for the network traffic recorded in the previous five minutes.

The maximum file size for a log file is 75 MB. If the log file reaches the file size limit within the 5-minute period, the log stops adding records to it, publishes it to the Amazon S3 bucket, and then creates a new log file.

A single log file contains interleaved entries with multiple 5-tuple records. To see all the log files for your firewall, look for entries aggregated by the firewall name and your account ID.

Log files are saved in the specified Amazon S3 bucket using a folder structure that's determined by the log's ID, Region, Network Firewall log type, and the date. The bucket folder structure uses the following format:

```
s3-bucket-name/optional-s3-bucket-prefix/AWSLogs/aws-account-id/network-firewall/log-type/Region/firewall-name/timestamp/
```

Similarly, the log file name is determined by the flow log's ID, Region, and the date and time it was created. File names use the following format:

```
aws-account-id_network-firewall_log-type_Region_firewall-name_timestamp_hash.log.gz
```

In the specification of the folder and file name, the following apply:

- The log type is either `alert` or `flow`.
- The timestamp uses the `YYYYMMDDTHHmmZ` format.
- If you don't provide a specification for the S3 bucket prefix, the log file bucket folder structure will be similar to the following:

  ```
  s3-bucket-name/AWSLogs/aws-account-id
  ```

- If you specify slash (`/`) for the S3 bucket prefix, or provide a prefix that begins with a slash, the log file bucket folder structure will contain a double slash (`//`), like the following for a prefix set to a single slash:

  ```
  s3-bucket-name//AWSLogs/aws-account-id
  ```

The following shows an example flow log file in Amazon S3 for AWS account `11111111111`, firewall name `test-firewall`, bucket name `s3://DOC-EXAMPLE-BUCKET`, and bucket prefix `flow-logs`.

```
s3://DOC-EXAMPLE-BUCKET/flow-logs/AWSLogs/11111111111/network-firewall/flow/us-
east-1/test-firewall/2020/10/01/19/11111111111_network-firewall_flow_us-east-1_test-
firewall_202010011920_44442222.log.gz
```

Permissions to publish logs to Amazon S3

You must have the following permissions settings to configure your firewall to send logs to Amazon S3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "FirewallLogging"
        },
        {
            "Sid": "FirewallLoggingS3",
            "Action": [
                "s3:PutBucketPolicy",
                "s3:GetBucketPolicy"
            ],
            "Resource": [
                Amazon S3 bucket ARN"
            ],
            "Effect": "Allow"
        }
    ]
}
```

By default, Amazon S3 buckets and the objects that they contain are private. Only the bucket owner can access the bucket and the objects stored in it. The bucket owner, however, can grant access to other resources and users by writing an access policy.

If the user creating the log owns the bucket, the service automatically attaches the following policy to the bucket to give the log permission to publish logs to it:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryWrite",
            "Effect": "Allow",
            "Principal": {"Service": "delivery.logs.amazonaws.com"},
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::bucket-name/optional-folder/AWSLogs/account-id/*",
            "Condition": {"StringEquals": {"s3:x-amz-acl": "bucket-owner-full-control"}}
        },
        {
            "Sid": "AWSLogDeliveryAclCheck",
```

```
            "Effect": "Allow",
            "Principal": {"Service": "delivery.logs.amazonaws.com"},
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::bucket-name"
        }
    ]
}
```

If the user creating the log doesn't own the bucket, or doesn't have the `GetBucketPolicy` and `PutBucketPolicy` permissions for the bucket, the log creation fails. In this case, the bucket owner must manually add the preceding policy to the bucket and specify the log creator's AWS account ID. For more information, see How Do I Add an S3 Bucket Policy? in the *Amazon Simple Storage Service User Guide*. If the bucket receives logs from multiple accounts, add a `Resource` element entry to the `AWSLogDeliveryWrite` policy statement for each account.

For example, the following bucket policy allows AWS accounts `111122223333` and `444455556666` to publish logs to a folder named `flow-logs` in a bucket named `DOC-EXAMPLE-BUCKET1`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryWrite",
            "Effect": "Allow",
            "Principal": {"Service": "delivery.logs.amazonaws.com"},
            "Action": "s3:PutObject",
            "Resource": [
             "arn:aws:s3:::log-bucket/flow-logs/AWSLogs/111122223333/*",
             "arn:aws:s3:::log-bucket/flow-logs/AWSLogs/444455556666/*"
             ],
            "Condition": {"StringEquals": {"s3:x-amz-acl": "bucket-owner-full-control"}}
        },
        {
            "Sid": "AWSLogDeliveryAclCheck",
            "Effect": "Allow",
            "Principal": {"Service": "delivery.logs.amazonaws.com"},
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
        }
    ]
}
```

Amazon S3 log file access

In addition to the required bucket policies, Amazon S3 uses access control lists (ACLs) to manage access to the log files created by a Network Firewall log. By default, the bucket owner has `FULL_CONTROL` permissions on each log file. The log delivery owner, if different from the bucket owner, has no permissions. The log delivery account has `READ` and `WRITE` permissions. For more information, see Access Control List (ACL) Overview in the *Amazon Simple Storage Service User Guide*.

The log files are compressed. If you open the files using the Amazon S3 console, Amazon S3 decompresses the log records and displays them. If you download the log files, you must decompress them to view the records.

Amazon CloudWatch Logs

To send logs to Amazon CloudWatch Logs, you create a CloudWatch Logs log group. When you enable logging in Network Firewall, you provide the log group name. After you enable logging for your firewall, AWS Network Firewall delivers logs to the CloudWatch Logs log group in log streams. Each log stream contains an hour of log records.

You can use any name for your CloudWatch Logs log group. Configure the log group in the same Region as the firewall and using the same account as you use to manage the firewall.

For information about configuring a CloudWatch Logs log group, see Working with Log Groups and Log Streams.

When you configure your Network Firewall firewall to send logs to that log group, the resulting log streams have the following naming format:

```
/aws/network-firewall/log-type/firewall-name_YYYY-MM-DD-HH
```

In the specification, the log type is either `alert` or `flow`.

The following shows an example log stream created on October 1, 2020, at 5 pm for alert logging for firewall `test-firewall`.

```
/aws/network-firewall/alert/test-firewall_2020-10-01-17
```

Permissions to publish logs to CloudWatch Logs

You must have the following permissions settings to configure your firewall to send logs to a CloudWatch Logs log group.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "FirewallLogging"
        },
        {
            "Sid": "FirewallLoggingCWL",
            "Action": [
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": [
                "CloudWatch Logs log group ARN"
            ],
            "Effect": "Allow"
        }
    ]
}
```

Amazon Kinesis Data Firehose

To send logs to Amazon Kinesis Data Firehose, you first need to set up a Firehose delivery stream. As part of that process, you choose a destination for storing your logs. After you enable logging for your firewall,

AWS Network Firewall delivers logs to the destination through the HTTPS endpoint of Amazon Kinesis Data Firehose. One AWS Network Firewall log corresponds to one Amazon Kinesis Data Firehose record.

Configure an Amazon Kinesis Data Firehose delivery stream for your firewall as follows.

- Create it using the same account as you use to manage the firewall.
- Create it in the same Region as the firewall.
- Configure it for direct put, which allows applications to access the delivery stream directly. In the Amazon Kinesis Data Firehose console, for the delivery stream **Source** setting, choose **Direct PUT or other sources**. Through the API, set the delivery stream property `DeliveryStreamType` to `DirectPut`.

For information about how to create an Amazon Kinesis Data Firehose delivery stream and review the stored logs, see Creating an Amazon Kinesis Data Firehose delivery stream and What is Amazon Kinesis Data Firehose?

When you successfully enable logging to an Amazon Kinesis Data Firehose data stream, Network Firewall creates a service linked role with the necessary permissions to write logs to it. For more information, see Using service-linked roles for Network Firewall (p. 75) For more information about service-linked roles, see Using service-linked roles for Network Firewall (p. 75).

Permissions to publish logs to Amazon Kinesis Data Firehose

You must have the following permissions to configure your firewall to send logs to an Amazon Kinesis Data Firehose delivery stream.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "FirewallLogging"
        },
        {
            "Sid": "FirewallLoggingFH",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Sid": "FirewallLoggingFH",
            "Action": [
                "firehose:TagDeliveryStream"
            ],
            "Resource": "Amazon Kinesis Data Firehose delivery stream ARN",
            "Effect": "Allow"
        }
    ]
}
```

Logging with server-side encryption and customer-provided keys

If your logging destination uses server-side encryption with keys that are stored in AWS Key Management Service (SSE-KMS) and you use a customer managed key (KMS key), you must give Network Firewall permission to use your KMS key. To do this, you add a key policy to the KMS key for your chosen destination to permit Network Firewall logging to write your log files to the destination.

Policy for an Amazon S3 bucket

Add the following key policy to your KMS key to allow Network Firewall to log to your Amazon S3 bucket.

```
{
    "Sid": "Allow Network Firewall to use the key",
    "Effect": "Allow",
    "Principal": {
        "Service": [
            "delivery.logs.amazonaws.com"
        ]
    },
    "Action": "kms:GenerateDataKey*",
    "Resource": "*"
}
```

> **Note**
> Network Firewall supports encryption with Amazon S3 buckets for key type Amazon S3 key (SSE-S3) and for AWS Key Management Service (SSE-KMS) AWS KMS keys. Network Firewall doesn't support encryption for AWS Key Management Service keys that are managed by AWS.

Policy for a CloudWatch Logs log group

For a CloudWatch Logs log group, the service principal requires access to the logs for the Region. This is the same as for all encrypted CloudWatch Logs log streams. For more information about log data encryption in CloudWatch Logs, see Encrypt Log Data in CloudWatch Logs Using AWS KMS.

Add the following key policy to your KMS key to allow Network Firewall to log to your CloudWatch Logs log group.

```
{
    "Effect": "Allow",
    "Principal": {
        "Service": "logs.{region}.amazonaws.com"
    },
    "Action": [
        "kms:Encrypt*",
        "kms:Decrypt*",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:Describe*"
    ],
    "Resource": "*"
}
```

Policy for a Kinesis Data Firehose delivery stream

For Kinesis Data Firehose delivery streams, you allow the service principal to generate keys so that it can put the logging records.

Add the following key policy to your KMS key to allow Network Firewall to log to your Kinesis Data Firehose delivery stream.

```
{
    "Sid": "Allow Network Firewall logs to use the key",
    "Effect": "Allow",
    "Principal": {
        "Service": [
            "delivery.logs.amazonaws.com"
        ]
    },
    "Action": "kms:GenerateDataKey*",
    "Resource": "*"
}
```

Updating a firewall's logging configuration

To update your firewall's logging configuration through the AWS Management Console, use the procedure in this section. For the API, see the API action, `UpdateLoggingConfiguration`.

> **Note**
> Firewall logging is only available for traffic that you forward to the stateful rules engine. You forward traffic to the stateful engine through stateless rule actions and stateless default actions in the firewall policy. For information about these actions settings, see Stateless default actions in your firewall policy (p. 35) and Rule actions in AWS Network Firewall (p. 55).

To update a firewall's logging configuration through the console

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Network Firewall**, choose **Firewalls**.
3. In the **Firewalls** page, choose the name of the firewall that you want to edit. This takes you to the firewall's details page.
4. Choose the tab **Firewall details**, then in the **Logging** section, choose **Edit**.
5. Adjust the **Log type** selections as needed. You can configure logging for alert and flow logs.
   - **Alert** – Sends logs for traffic that matches any stateful rule whose action is set to `Alert` or `Drop`. For more information about stateful rules and rule groups, see Rule groups in AWS Network Firewall (p. 39).
   - **Flow** – Sends logs for all network traffic that the stateless engine forwards to the stateful rules engine.
6. For each selected log type, choose the destination type, then provide the information for the logging destination that you prepared following the guidance in Firewall logging destinations (p. 85).
7. Choose **Save** to save your changes and return to the firewall's detail page.

Logging calls to the AWS Network Firewall API with AWS CloudTrail

AWS Network Firewall is integrated with AWS CloudTrail, a service that provides a record of API calls to Network Firewall by a user, role, or an AWS service. CloudTrail captures all API calls for Network Firewall as events. The calls captured include calls from the Network Firewall console and code calls to the Network Firewall API operations. If you create a trail, you can enable continuous delivery of CloudTrail

events to an Amazon S3 bucket, including events for Network Firewall. 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 information including the request that was made to Network Firewall, the IP address from which the request was made, who made the request, and when the request was made.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

AWS Network Firewall information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Network Firewall, it's 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 Network Firewall, create a trail. A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS 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

All Network Firewall actions are logged by CloudTrail. These actions are documented in the Actions section of the AWS Network Firewall API Reference. For example, calls to the actions `CreateFirewall`, `ListFirewalls`, and `DeleteFirewall` 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 AWS Identity and Access Management (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.

CloudTrail log file examples

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 aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following are examples of CloudTrail log entries for Network Firewall operations.

Example: CloudTrail log entry for `CreateFirewall`

```
{
```

```
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLEPrincipalId",
    "arn": "arn:aws:sts::444455556666:assumed-role/Admin/EXAMPLE",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "EXAMPLEPrincipalId",
        "arn": "arn:aws:iam::444455556666:role/Admin",
        "accountId": "444455556666",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2020-08-13T03:07:52Z"
      }
    }
  },
  "eventTime": "2020-08-13T03:07:53Z",
  "eventSource": "network-firewall.amazonaws.com",
  "eventName": "CreateFirewall",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.4",
  "userAgent": "aws-cli/1.18.117 Python/3.6.10
 Linux/4.9.217-0.1.ac.205.84.332.metal1.x86_64 botocore/1.17.40",
  "requestParameters": {
    "firewallName": "firewall01",
    "firewallPolicyArn": "arn:aws:network-firewall:us-west-2:444455556666:firewall-policy/
policy01",
    "vpcId": "vpc-11112222",
    "subnetMappings": [
      {
        "subnetId": "subnet-44443333",
        "requestedCapacity": "10"
      }
    ],
    "deleteProtection": false
  },
  "responseElements": {
    "firewall": {
      "firewallName": "firewall01",
      "firewallArn": "arn:aws:network-firewall:us-west-2:444455556666:firewall/firewall01",
      "firewallPolicyArn": "arn:aws:network-firewall:us-west-2:444455556666:firewall-
policy/policy01",
      "vpcId": "vpc-11112222",
      "subnetMappings": [
        {
          "subnetId": "subnet-44443333",
          "requestedCapacity": "10"
        }
      ],
      "deleteProtection": false
    },
    "firewallStatus": {
      "status": "PROVISIONING",
      "configurationSyncStateSummary": "PENDING"
    }
  },
  "requestID": "43a8cad0-68b6-45d2-b6f3-28cf0e195d47",
  "eventID": "7d575a14-ec3f-43c8-8735-eaadd21fd9d1",
  "readOnly": false,
  "eventType": "AwsApiCall",
```

```
  "recipientAccountId": "444455556666"
}
```

Example: CloudTrail log entry for `ListFirewalls`

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLEPrincipalId",
    "arn": "arn:aws:sts::444455556666:assumed-role/Admin/EXAMPLE",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "EXAMPLEPrincipalId",
        "arn": "arn:aws:iam::444455556666:role/Admin",
        "accountId": "444455556666",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2020-08-13T03:07:55Z"
      }
    }
  },
  "eventTime": "2020-08-13T03:07:55Z",
  "eventSource": "network-firewall.amazonaws.com",
  "eventName": "ListFirewalls",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.4",
  "userAgent": "aws-cli/1.18.117 Python/3.6.10
 Linux/4.9.217-0.1.ac.205.84.332.metal1.x86_64 botocore/1.17.40",
  "requestParameters": {
    "maxResults": 10
  },
  "responseElements": null,
  "requestID": "1ac1567a-fa84-49ac-b5aa-6016052ad646",
  "eventID": "79b95fd6-a288-49b1-a907-b61ed99b94c0",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "recipientAccountId": "444455556666"
}
```

Example: CloudTrail log entry for `DeleteFirewall`

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLEPrincipalId",
        "arn": "arn:aws:sts::444455556666:assumed-role/Admin/EXAMPLE",
        "accountId": "444455556666",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLEPrincipalId",
                "arn": "arn:aws:iam::444455556666:role/Admin",
                "accountId": "444455556666",
```

```
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2020-08-19T16:09:29Z"
            }
        }
    },
    "eventTime": "2020-08-19T16:18:43Z",
    "eventSource": "network-firewall.amazonaws.com",
    "eventName": "DeleteFirewall",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "198.51.100.190",
    "userAgent": "Apache-HttpClient/UNAVAILABLE (Java/1.8.0_232)",
    "requestParameters": {
        "firewallArn": "arn:aws:network-firewall:us-west-2:444455556666:firewall/
DeleteMeFast"
    },
    "responseElements": {
        "firewall": {
            "firewallName": "DeleteMeFast",
            "firewallArn": "arn:aws:network-firewall:us-west-2:444455556666:firewall/
DeleteMeFast",
            "firewallPolicyArn": "arn:aws:network-firewall:us-west-2:444455556666:firewall-
policy/123",
            "vpcId": "vpc-11112222",
            "subnetMappings": [
                {
                    "subnetId": "subnet-99990000",
                    "requestedCapacity": "14"
                },
                {
                    "subnetId": "subnet-77776666",
                    "requestedCapacity": "12"
                }
            ],
            "deleteProtection": true,
            "description": "HIDDEN_DUE_TO_SECURITY_REASONS"
        },
        "firewallStatus": {
            "status": "DELETING",
            "configurationSyncStateSummary": "PENDING",
            "syncStates": {
                "us-west-2c": {
                    "attachment": {
                        "subnetId": "subnet-99990000",
                        "networkInterfaceId": "eni-01e59ab6f6064c453",
                        "status": "SCALING"
                    },
                    "config": {}
                },
                "us-west-2d": {
                    "attachment": {
                        "subnetId": "subnet-77776666",
                        "networkInterfaceId": "eni-04c3ac8c04076ed36",
                        "status": "SCALING"
                    },
                    "config": {}
                }
            }
        }
    },
    "requestID": "299b886e-23da-4c77-8beb-0853a0a08bcf",
    "eventID": "142b089a-8aca-4183-8326-5ff32a38876e",
    "readOnly": false,
```

```
    "eventType": "AwsApiCall",
    "recipientAccountId": "444455556666"
}
```

AWS Network Firewall metrics in Amazon CloudWatch

You can monitor AWS Network Firewall using CloudWatch, which collects raw data and processes it into readable, near real-time metrics. CloudWatch stores your metrics for 15 months, so that you can access historical information for added perspective on how your web application or service is performing. You can also set alarms that watch for certain thresholds, and send notifications or take actions when those thresholds are met. For more information, see the Amazon CloudWatch User Guide.

Use the following procedures to view the metrics for Network Firewall.

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. The CloudWatch namespace for Network Firewall is AWS/NetworkFirewall.

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, choose **Metrics**.
3. On the **All metrics** tab, choose the Region and then choose `AWS/NetworkFirewall`.

To view metrics using the AWS CLI

- For Network Firewall, at a command prompt use the following command:

```
aws cloudwatch list-metrics --namespace "AWS/NetworkFirewall"
```

AWS Network Firewall metrics

The AWS/NetworkFirewall namespace includes the following metrics.

Metric	Description
`DroppedPackets`	Number of packets dropped by the Network Firewall firewall. Reporting criteria: There is a nonzero value. Valid statistics: Sum
`Packets`	Number of packets inspected for a firewall policy or stateless rulegroup for which a custom action is defined. This metric is only used for the dimension `CustomAction`. Reporting criteria: There is a nonzero value. Valid statistics: Sum

Metric	Description
`PassedPackets`	Number of packets that the Network Firewall firewall allowed through to their destinations. Reporting criteria: There is a nonzero value. Valid statistics: Sum
`ReceivedPacketCount`	Number of packets received by the Network Firewall firewall. Reporting criteria: There is a nonzero value. Valid statistics: Sum

AWS Network Firewall dimensions

Network Firewall can use the following dimension combinations to categorize your metrics:

Dimension	Description
`AvailabilityZone`	Availability Zone in the Region where the Network Firewall firewall is active.
`CustomAction`	Dimension for a publish metrics custom action that you defined. You can define this for a rule action in a stateless rule group or for a stateless default action in a firewall policy.
`Engine`	Rules engine that processed the packet. The value for this is either `Stateful` or `Stateless`.
`FirewallName`	Name that you specified for the Network Firewall firewall.

Tagging AWS Network Firewall resources

A *tag* is a custom attribute label that you assign or that AWS assigns to an AWS resource. Each tag has two parts:

- A *tag key*, for example `CostCenter`, `Environment`, or `Project`. Tag keys are case sensitive.
- An optional *tag value*, for example, `111122223333` or `Production`. Omitting the tag value is the same as using an empty string. Tag values are case sensitive.

You can use tags to do the following:

- Identify and organize your AWS resources. Many AWS services support tagging, so you can assign the same tag to resources from different services to indicate that the resources are related. For example, you could assign the same tag to an Amazon Virtual Private Cloud VPC that you assign to an firewall and firewall policy in AWS Network Firewall.
- Track your AWS costs. To do this, you activate tags on the AWS Billing and Cost Management dashboard. AWS uses the tags to categorize your costs and deliver a monthly cost allocation report to you. For more information, see Use cost allocation tags in the *AWS Billing and Cost Management User Guide*.

The following sections provide more information about tags for AWS Network Firewall.

Supported resources in Network Firewall

The following resources in Network Firewall support tagging:

- Firewalls
- Firewall policies
- Rule groups

For information about adding and managing tags, see Managing tags (p. 100).

Tag naming and usage conventions

The following basic naming and usage conventions apply to using tags with Network Firewall resources:

- Each resource can have a maximum of 50 tags.
- For each resource, each tag key must be unique, and each tag key can have only one value.
- The maximum tag key length is 128 Unicode characters in UTF-8.
- The maximum tag value length is 256 Unicode characters in UTF-8.
- Allowed characters are letters, numbers, spaces representable in UTF-8, and the following characters: **. : + = @ _ / -** (hyphen). Amazon EC2 resources allow any characters.

- Tag keys and values are case sensitive. As a best practice, decide on a strategy for capitalizing tags, and consistently implement that strategy across all resource types. For example, decide whether to use `Costcenter`, `costcenter`, or `CostCenter`, and use the same convention for all tags. Avoid using similar tags with inconsistent case treatment.
- The `aws:` prefix is prohibited for tags; it's reserved for AWS use. You can't edit or delete tag keys or values with this prefix. Tags with this prefix do not count against your tags per resource quota.

Managing tags

For ease of use and best results, use the Tag Editor in the AWS Resource Groups console. It provides a central, unified way to create and manage your tags. For more information, see Working with Tag Editor.

You can also use AWS Network Firewall to apply tags while you are creating and managing your Network Firewall firewalls, firewall policies, and rule groups.

AWS Network Firewall quotas

AWS Network Firewall is subject to the following quotas (formerly referred to as limits). These quotas are the same for all AWS Regions in which Network Firewall is available. Each Region is subject to these quotas individually. The quotas are not cumulative across Regions.

Network Firewall has the following default quotas on the maximum number of entities you can have per account per Region. You can request a modification to these quotas by creating a support case for a service limit increase at the AWS Support Center.

Resource	Default quota per account per Region
Maximum number of firewalls.	5
Maximum number of firewall policies.	20
Maximum number of stateful rule groups.	50
Maximum number of stateless rule groups.	50

Network Firewall has the following quotas that can't be changed.

Resource	Quota per account per Region
Maximum size of a Suricata-compatible rules string for a rule group, in bytes.	2,000,000
Maximum stateful rule group capacity. For more information, see Rule group capacity in AWS Network Firewall (p. 54).	30,000
Maximum number of stateful rule groups per firewall policy.	10
Maximum number of stateful rules per firewall policy. This is the total across all rule groups that are referenced by the policy.	30,000
Maximum stateless rule group capacity. For more information, see Rule group capacity in AWS Network Firewall (p. 54).	30,000
Maximum number of stateless rule groups per firewall policy.	10
Maximum number of stateless rules per firewall policy. This is the total across all rule groups that are referenced by the policy.	10,000
Required number of firewall policies per firewall.	1
Maximum number of firewalls that can use the same firewall policy.	1,000
Maximum number of firewall policies that can use the same rule group.	1,000

Using the AWS Network Firewall REST API

This section describes how to make requests to the Network Firewall API for creating and managing firewalls in Network Firewall. This section covers the components of requests, the content of responses, and how to authenticate requests.

For general guidance on accessing the AWS APIs, see the AWS APIs in the *AWS General Reference*.

> **Note**
> If you use a programming language that has an AWS SDK, use the SDK rather than trying to work your way through the APIs. The SDKs make authentication simpler, integrate more easily with your development environment, and provide easy access to Network Firewall commands. For more information about the AWS SDKs, see Setting up tool access (p. 20) in the topic Setting up (p. 18).

Topics

Making HTTPS requests to AWS Network Firewall

Network Firewall requests are HTTPS requests, as defined by RFC 2616. Like any HTTP request, a request to Network Firewall 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

Network Firewall requires the following information in the header of an HTTP request.

Host (Required)

The endpoint that specifies where your resources are created. You can find the various endpoints in AWS service endpoints. For example, the value of the `Host` header for Network Firewall for a CloudFront distribution is `network-firewall.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, Network Firewall ignores any `Date` header when authenticating the request.

The timestamp must be within 15 minutes of the AWS system time when the request is received. If it isn't, the request fails with the `RequestExpired` error code to prevent someone else from replaying your requests.

Authorization (Required)

The information required for request authentication. For more information about constructing this header, see Authenticating requests (p. 104).

X-Amz-Target (Required)

The operation, provided as a concatenation of the following values:

- `NetworkFirewall_`
- `The API version without punctuation`
- `A period (.)`
- `The name of the operation`

Example:

`NetworkFirewall_20201112.CreateFirewall`

Content-Type (Conditional)

The type and version of the content. Specify the version of JSON, as shown in the following example:

```
Content-Type: application/x-amz-json-1.0
```

Condition: Required for POST requests.

Content-Length (Conditional)

The 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 firewall in Network Firewall:

```
POST / HTTP/1.1
Host: network-firewall.amazonaws.com:443
X-Amz-Date: 20151007T174952Z
Authorization: AWS4-HMAC-SHA256
               Credential=AKIAIOSFODNN7EXAMPLE/20151007/us-east-2/network-firewall/
aws4_request,
               SignedHeaders=host;x-amz-date;x-amz-target,
               Signature=145b1567ab3c50d929412f28f52c45dbf1e63ec5c66023d232a539a4afd11fd9
X-Amz-Target: NetworkFirewall_20201112.CreateFirewall
Accept: */*
Content-Type: application/x-amz-json-1.0; charset=UTF-8
Content-Length: 231
Connection: Keep-Alive
```

HTTP request body

Many Network Firewall API actions require you to include JSON-formatted data in the body of the request.

HTTP responses

All Network Firewall 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 Network Firewall that uniquely identifies your request, for example, `K2QH8DNOU907N97FNA2GDLL8OBVV4KQNSO5AEMVJF66Q9ASUAAJG`. If you have a problem with Network Firewall, 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 Network Firewall responded, for example, Wed, 07 Oct 2019 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
- A `Content-Type` header
- 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 Network Firewall API. In addition, the SDKs integrate easily with your development environment and provide easy access to related commands.

Network Firewall requires 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.

Network Firewall supports authentication using AWS Signature Version 4. Follow the process for signing your request at see the Signing AWS requests with Signature Version 4 in the *AWS General Reference*.

After receiving your request, Network Firewall 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, Network Firewall processes the request. If not, Network Firewall rejects the request.

Resources

The following related resources can help you as you work with this service.

AWS resources

Amazon Web Services provides the following resources specific to AWS Network Firewall.

- **Network Firewall discussion forum** – A community-based forum for developers to discuss technical questions related to Network Firewall.
- **Network Firewall product information** – The primary web page for information about Network Firewall, including features and pricing.
- **Amazon Virtual Private Cloud product information** – The primary web page for information about Amazon VPC, including features and pricing.

Amazon Web Services provides the following general guides, forums, and other resources.

- **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 for AWS Network Firewall

The following table describes important changes to this documentation.

update-history-change	update-history-description	update-history-date
Optional strict evaluation order for Suricata compatible stateful rule groups (p. 42)	This release adds support for strict ordering for stateful rule groups. Using strict ordering, stateful rule groups are evaluated in the exact order in which you provide them in the firewall policy.	October 1, 2021
Expanded availability of AWS managed policy (p. 78)	Network Firewall expanded the availability of the managed policy `AWSNetworkFirewallServiceRolePolicy` to AWS GovCloud (US) Regions.	June 24, 2021
Increased stateless rule group capacity (p. 101)	The capacity for stateless rule groups is increased from 10,000 to 30,000.	June 10, 2021
Reorganized stateful rule groups sections and expanded examples (p. 41)	Domain list rule groups and the simple (5-tuple) rule groups provide easy entry forms for Suricata compatible IPS rules, and the documentation didn't indicate this. Reorganized stateful rule group sections, clarified the information, and added examples showing the correlation between the easy entry forms and the resulting Suricata compatible IPS rules.	April 28, 2021
JA3 keywords support (p. 42)	JA3 keywords are now supported by Network Firewall.	April 28, 2021
First release of AWS Network Firewall (p. 1)	Network Firewall is now available to provide firewall protection for your Amazon Virtual Private Cloud VPCs.	November 16, 2020

AWS glossary

For the latest AWS terminology, see the AWS glossary in the *AWS General Reference*.