AWS Interconnect User Guide

AWS Interconnect

AWS Interconnect: AWS Interconnect User Guide

Table of Contents

What is AWS Interconnect?

What is AWS Interconnect?

AWS Interconnect is a family of managed private connectivity services that enables you to create high-speed connections from your locations to AWS and between cloud environments.

With AWS Interconnect, you no longer need to configure physical or virtual routers, order cross-connects, or manage BGP peering. Through a simplified process you select your AWS region, your required network capacity, and your preferred provider.

AWS and the provider will quickly provision and configure your requested capacity on redundant network devices. They present it to you as a single object called an *interconnect*.

AWS Interconnect includes two offerings:

- Interconnect – multicloud: Connect your AWS VPCs directly to VPCs in other public clouds.
- Interconnect – last mile: Connect your branch offices, data centers, and remote locations to AWS through qualified delivery partners' existing last-mile networks.

Advantages of AWS Interconnect

Simplified Network Architecture

With AWS Interconnect, your traffic is transported on the AWS global backbone until it is handed off to the provider directly. This eliminates the need for complex on-premises routing or intermediate network hops.

Fast Provisioning and Scaling

New Interconnects are typically provisioned and configured within minutes. Once live, you can increase or decrease bandwidth directly through the AWS Console without reprovisioning connections or engaging support. This elastic model lets you scale up for peak workloads and scale back down to control costs.

Fully managed service

AWS and your provider manage all aspects of the physical network infrastructure and configuration. This includes provisioning network devices, configuring VLANs and BGP sessions, and handling ongoing maintenance. You focus on your applications while AWS handles the underlying connectivity infrastructure.

Built-in maximum resiliency

Every Interconnect is provisioned across redundant network devices spanning at least two physically distinct facilities with independent power and networking. This architecture eliminates single points of failure at the device, cross-connect, and facility level. Multicloud and last mile connections use a four-connection model with Equal-Cost Multi-Path (ECMP) load balancing, ensuring at least one link remains operational during planned maintenance.

Region Availability

AWS Interconnect – last mile

AWS Interconnect – last mile initially launches with Lumen in us-east-1 (N. Virginia). You can create a last mile connection from the NJ sites in us-east-1 to any AWS region globally, and connect from anywhere in the continental United States via the Lumen connectivity fabric. Additional partners and regions will be added over time.

AWS Interconnect – Multicloud

Interconnect - multicloud is supported in the following AWS and Google Cloud Regions:

- AWS US East (N. Virginia) us-east-1 – Google Cloud N. Virginia (us-east4)
- AWS US West (N. California) us-west-1 – Google Cloud Los Angeles (us-west2)
- AWS US West (Oregon) us-west-2 – Google Cloud Oregon (us-west1)
- AWS Europe (London) eu-west-2 – Google Cloud London (europe-west2)
- AWS Europe (Frankfurt) eu-central-1 – Google Cloud Frankfurt (europe-west3)

How AWS Interconnect works

AWS Interconnect is designed to connect your private networks in AWS with your private networks in remote locations or other cloud environments. You don't need to think about physical networking devices or configure routing protocols. The service is also designed to provision capacity quickly. It follows our standards for security and maximum network resiliency.

To achieve these design goals, AWS and its partners have pre-provisioned capacity in each of the supported regions and locations. This capacity spans multiple network devices distributed across at least two physical buildings. These buildings have independent power and networking.

All connections between the AWS network devices and the adjacent partner devices are encrypted by default. The encryption uses industry standard IEEE 802.1AE MAC Security (MACsec). The devices are configured to transmit customer traffic only if the encryption session is active.

AWS Interconnect concepts

Interconnect

The managed connectivity object provisioned between AWS and a provider (cloud service provider or last-mile provider). Each Interconnect is presented as a single logical object in your AWS account, abstracting the underlying redundant infrastructure.

Attach point

A logical identifier that anchors the Interconnect on each side of the connection. On AWS, the attach point is always the Direct Connect Gateway. On the remote side, the attach point varies by offering: a CSP router (Multicloud) or a partner-side network identifier (last mile).

Direct Connect Gateway

A logical, highly-available, globally distributed object that serves as the attach point for Region or Local Zone-based AWS networking services such as Virtual Private Gateways, Transit Gateways, and Cloud WAN, as well as AWS Interconnect.

Activation Key

A token generated during the connection creation process. It is shared between parties (AWS and provider) to authorize and complete the provisioning of a new Interconnect, ensuring that both sides validate the request before resources are committed.

Create/Accept flows

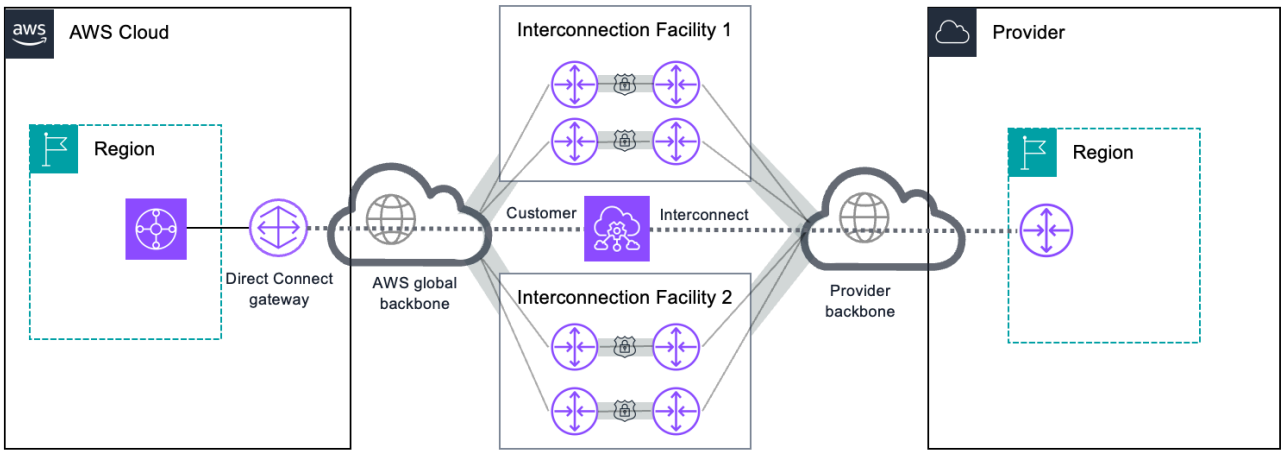
The process to create a new Interconnect has two main actions: one party initiates a Create action, generating an Activation Key. The other party uses that key to perform the Accept action, triggering automated provisioning on both sides.

Built-in resiliency and security

AWS Interconnect uses redundant infrastructure to provide maximum network resiliency. The service automatically provisions multiple logical connections spanning multiple network devices across two or more physical facilities. All physical connections between AWS routers and partner devices are secured using MACsec encryption. The customer sees only a single abstracted Interconnect object in their console, while the underlying infrastructure ensures business continuity during device or facility failures.

The following diagram shows the high-level physical infrastructure used to provide AWS Interconnect. The customer Interconnect is represented by the grey dotted line attachment. This attachment is created directly between the attach points on both sides. The customer sees only the abstracted object in their console. In this example, the customer is using a Transit Gateway attached to a Direct Connect gateway. This gateway is used as the attach point on the AWS side.

At the infrastructure level, AWS and the provider have provisioned multiple logical connections. These connections span multiple network devices across two physical facilities. All the physical connections between the AWS routers and the provider routers in each facility are secured using MACsec.

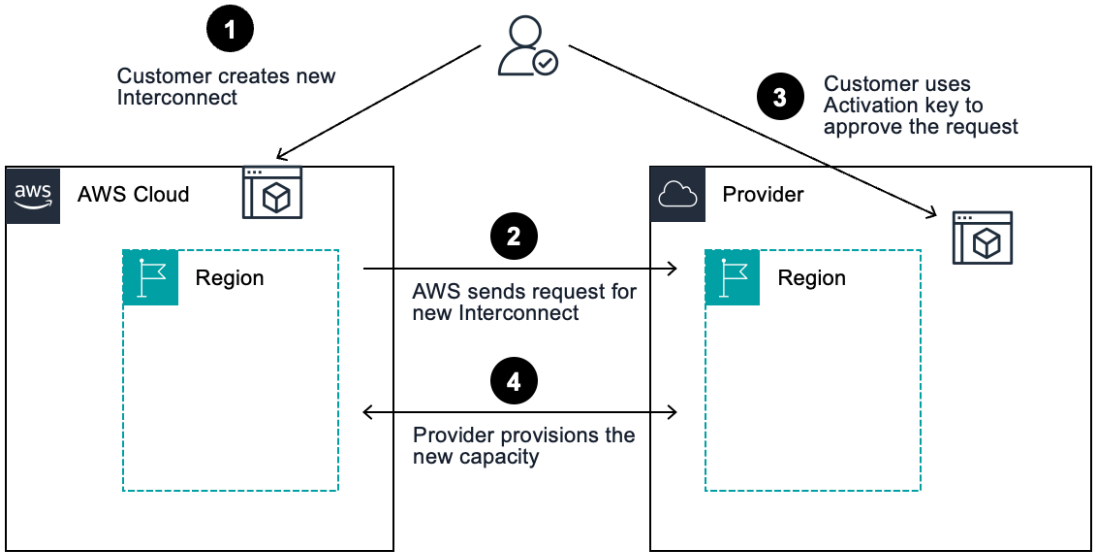


Interconnect Creation Process at High-Level

In the following example diagram, a customer begins the process by creating a new Interconnect using the AWS Console (1). AWS receives the request. AWS provides the customer with the new Interconnect activation key. AWS submits the request to the provider (2).

The provider receives the request and waits for confirmation from the customer. The customer then uses the activation key to approve the request (3).

AWS and the provider then begin the provisioning process. No further customer interaction is needed (4). The process is completed with the successful creation of the new Interconnect. The Interconnect attaches to the Direct Connect gateway on the AWS side and to the provider's attach point on the remote side.



Supported configurations with AWS networking services

AWS Interconnect can connect to your VPCs through supported AWS networking services: Virtual private gateways, Transit Gateways, and Cloud WAN.

Virtual Private Gateways and Transit Gateways

Virtual gateways or Transit Gateways in a specific AWS Region, through a Direct Connect gateway, can only reach an Interconnect that provides connectivity to the paired Google Cloud Region. This Interconnect is considered "local" to that AWS Region. For example, a Transit Gateway in the AWS Region N. Virginia (us-east-1) can only reach an Interconnect that connects to the Google Cloud N. Virginia (us-east4) Region.

Cloud WAN

When using Cloud WAN you define the AWS Regions where your global network will have a Core Network Edge (CNE). Using the native Direct Connect attachment, any Cloud WAN CNE can reach any Interconnect globally that is attached to the same Direct Connect gateway.

Network health monitoring

All Interconnects include a single CloudWatch Network Synthetic Monitor at no extra cost. You can use this active synthetic probe to produce round trip latency and packet loss metrics. You can configure CloudWatch alarms on your set thresholds. Refer to the CloudWatch Network Synthetic Monitor user guide for configuration. Note that the Network Health Indicator feature is not yet supported with Interconnects. Latency and packet loss metrics are fully supported.

Bandwidth utilization monitoring

Your Interconnects provide a percentage utilization metric on CloudWatch. This metric displays the percentage of the capacity of your Interconnect that your traffic is utilizing. This metric is designed to help you understand your usage patterns and increase or decrease your provisioned bandwidth as needed.

You can set up CloudWatch alarms on your desired thresholds to help you prevent potential congestion events due to an application consuming all of your provisioned network capacity on an Interconnect.

Getting started with AWS Interconnect - multicloud

Plan your network architecture

- Decide whether to use a Virtual private gateway, Transit Gateway, or Cloud WAN. Virtual private gateways and Transit Gateways are Regional networking services that can be used only with a multicloud Interconnect provisioned in the local interconnection point to Google Cloud serving that Region. Cloud WAN is a global networking service which can reach any Interconnect globally.
- Review your existing IP address allocations to ensure no conflicts.
- Create a new Direct Connect gateway or repurpose an existing one for use with your new multicloud Interconnect.

Create your first multicloud Interconnect starting from the AWS Console

1. Go to the AWS Direct Connect Console and navigate to AWS Interconnect on the left side navigation menu.
2. Select **Create new multicloud Interconnect**.
3. Select Google Cloud as your provider.
4. Select your source AWS Region where your workload is located and destination region in Google Cloud.
5. Provide a name or description for your new interconnect, select your required bandwidth (limited to 1Gbps during Public Preview), specify an existing Direct Connect gateway to serve as the attach point for the new multicloud Interconnect, and provide the Google Cloud project ID. The project ID is a unique string that can be a combination of letters, numbers, and hyphens, between 6 and 30 characters in length. You can optionally apply a tag to your new interconnect. Choose **Next** when you have provided all the necessary information.
6. On the following screen, you can review the details of your new multicloud Interconnect. Choose **Finish** to request the new interconnect.
7. At this point AWS will request the creation of the new multicloud Interconnect to Google Cloud and display the activation key you will use to complete the process on Google Cloud.

8. To complete the creation process use the Activation key following the instructions on Google Cloud.
9. Once you have activated the new Interconnect on Google Cloud, the creation process will complete with the attachment of the new Interconnect to the Direct Connect gateway you specified.
10. Use the main AWS Interconnect view in the AWS Direct Connect Console to review a list of all your Interconnects.

Accepting a new multicloud Interconnect created from Google Cloud using the AWS Console

1. Go to the AWS Direct Connect Console and navigate to AWS Interconnect on the left side navigation menu.
2. Select **Accept multicloud Interconnect**.
3. Enter into the text field the Activation key generated on Google Cloud as part of create action and select **Next**.
4. Provide a name or description for your new interconnect. Specify an existing Direct Connect gateway to serve as the attach point for the new multicloud Interconnect. You can optionally apply a tag to your new interconnect. Choose **Next** to continue the accept action.
5. On the following screen, you can review the details of the new multicloud Interconnect that was requested from Google Cloud. Choose **Finish** to accept the new multicloud Interconnect.

Getting started with AWS Interconnect - last mile

This guide walks you through creating your first AWS Interconnect - last mile connection.

Prerequisites

Before creating a last mile Interconnect, ensure you have the following:

- An active AWS account with appropriate IAM permissions for AWS Direct Connect and AWS Interconnect.
- An existing Direct Connect gateway (or you can create a new one during setup) to serve as the attach point for your last mile Interconnect.
- An existing relationship with an Interconnect — last mile partner.
- Customer Premises Equipment (CPE) at your remote site connected to the partner's connectivity fabric.

Plan your network architecture

- Decide whether to use a Virtual Private Gateway, Transit Gateway, or Cloud WAN.
- Virtual Private Gateways and Transit Gateways are Regional networking services.
- Cloud WAN is a global networking service that can reach any Interconnect globally.
- Review your existing IP address allocations to ensure no conflicts.
- Create a new Direct Connect gateway or repurpose an existing one for use with your new last mile Interconnect.
- Determine your bandwidth requirements. Supported tiers: 1, 2, 5, 10, 25, 50, and 100 Gbps.

Create your first last mile Interconnect starting from the AWS Console

Use this workflow if you are an existing subscriber of an Interconnect — last mile partner.

1. Go to the AWS Direct Connect Console and navigate to **Last mile Interconnect** on the left side navigation menu.

2. Select **Create new last mile Interconnect**.
3. Select the participating partner you have a relationship with (for example: **Lumen**).
4. Select the **Interconnect Metro** closest to your physical location (for example, New York, Chicago, Seattle). Based on your metro selection, the console will display the **AWS Region** that has direct connectivity from that location. This is essentially where AWS and the partner have collocated Interconnect - last mile infrastructure. This determines the AWS edge location where your traffic enters the AWS network.
5. Provide a **name** or description for your new interconnect, select your required **bandwidth**, specify an existing **Direct Connect Gateway** (or create a new one during set up) to serve as the attach point for the new last mile Interconnect, and provide the **partner account ID**. The partner account ID may be an email address (for Lumen) or a unique alphanumeric string. You can optionally apply a tag to your new interconnect. Choose **Next** when you have provided all the necessary information.
6. On the following screen, you can review the details of your new last mile Interconnect. Choose **Finish** to request the new interconnect.
7. At this point AWS will request the creation of the new last mile Interconnect to your partner and display the activation key you will use to complete the process on the partner portal (for example, Lumen).
8. To complete the creation process use the **Activation key** following the instructions on the partner portal (for example, Lumen).
9. Once you have activated the new Interconnect on the partner portal, the creation process will complete with the attachment of the new Interconnect to the Direct Connect Gateway you specified.
10. Use the main AWS Interconnect view in the AWS Direct Connect Console to review a list of all your Interconnects.

Accepting a new last mile Interconnect created on the partner console using the AWS Console

1. Go to the AWS Direct Connect Console and navigate to **Last mile Interconnect** on the left side navigation menu.
2. Select **Accept last mile Interconnect**.

3. Enter into the text field the Activation key generated on the partner portal as part of create action and select **Next**.
4. Provide a name or description for your new interconnect. Specify an existing Direct Connect Gateway to serve as the attach point for the new last mile Interconnect. You can optionally apply a tag to your new interconnect. Choose **Next** to continue the accept action.
5. On the following screen, you can review the details of the new last mile Interconnect that was requested from the partner. Choose **Finish** to accept the new last mile Interconnect.

What Happens Next

After you submit your request:

- Automatic provisioning begins: The service automatically provisions four connections between four router-pairs across two distinct Interconnect - last mile sites.
- Network configuration: BGP peering, VLANs, and ASN assignments are configured automatically.
- MACsec encryption: 256-bit MACsec encryption is enabled by default on all connections.
- Monitoring enabled: End-to-end monitoring becomes available, including availability, health, latency, and packet loss metrics.
- Your connection typically becomes operational within the timeframe specified during the request process. You can monitor the provisioning status in the AWS Direct Connect Console.

Other considerations

- Last mile Interconnects support IPv4 and IPv6 address families.
- Last mile Interconnects use Border Gateway Protocol (BGP) for dynamic routing between your network and AWS. The service automatically establishes BGP sessions, configures AS numbers, and handles route advertisements. All BGP details are abstracted from you.
- The MTU for last mile Interconnects is set automatically to 8500 (Jumbo Frames enabled by default).
- MACsec encryption is enabled by default between AWS and partner devices at the Interconnect location.
- Multiple first-mile connectivity options are supported depending on what the partner offers, such as Ethernet and MPLS.

- You can attach a last mile Interconnect to an existing Direct Connect Gateway that already has Private Virtual Interfaces or Transit Virtual Interfaces attached to it.

Pricing for AWS Interconnect

> **Note**
>
> This page describes pricing for the AWS side of your Interconnect only. The service provider on the other side of your Interconnect determines their pricing independently of AWS.

> **Important**
>
> AWS pricing is subject to change. For the most current pricing information, see the pricing page for each service on aws.amazon.com/pricing. The pricing examples in this documentation are for illustration purposes only and may not reflect current pricing.

How AWS Interconnect Pricing Works

AWS Interconnect uses a single-fee pricing model based on two customer inputs: the selected bandwidth, and the geographic scope of the connectivity. Interconnect charges are computed on an hourly basis and there are no separate charges for the data transferred.

When you create an Interconnect, you configure the following options which are used to determine the hourly price:

1. An AWS Region. This determines the physical devices where your Interconnect will be provisioned and it will be considered your new Interconnect's local Region. For example, if you select the us-east-1 Region (N. Virginia), your Interconnect will be provisioned on the physical infrastructure closest to that Region.
2. The paired Region on the other cloud service provider in the case of AWS Interconnect - multicloud, or a specific metro area where your third-party provider connects to AWS, in the case of AWS Interconnect - last mile.
3. The required bandwidth, from the supported options for the specific Region and Provider combination. Supported speeds can vary across providers and regions (for example, 10 Gbps to Google Cloud).
4. The Direct Connect gateway (DXGW) that will serve as the attach point for your Interconnect on the AWS side.

> **Note**
>
> A Direct Connect gateway is a logical object. It is used as a global routing instance that distributes your routes between Regional endpoints and edge network devices. It operates outside of the data traffic path. It is globally distributed to all the edge network devices where your Interconnects and Direct Connect Virtual interfaces are provisioned. To learn more, review the AWS Direct Connect documentation.

For example, a 1 Gbps Interconnect associated only to a TGW in the local Region will be less expensive than a 1 Gbps Interconnect provisioned in Tokyo and configured to reach a Cloud WAN Core Network Edge in the AWS us-east-1 (N. Virginia) Region.

To review the price for a specific path, use the AWS Interconnect Pricing tool. Select the AWS Region where your workload resides and the Region where your Interconnect was provisioned to see that specific path's Tier.

Tiered pricing and provisioned capacity

Interconnect is priced using a Tier structure. Each specific AWS Region to Interconnect path has been assigned a Tier and each Tier has a unique price for every offered bandwidth. The following table illustrates how the Tiers increase as the global scope of the connectivity increases. It is provided for guidance purposes only and should not be considered as strict categories.

Tier	Connectivity scope	General description
Tier 1	Local	Endpoint in the local Region of your Interconnect
Tier 2	Regional	Endpoint in a Region in the same general geographi c area or across areas with widespread networking infrastructure
Tier 3	Continental	Endpoint in a Region farther across a continental or geographical area
Tier 4	Long-haul	Endpoint in a Region across a large geographical area or different continents

Tier	Connectivity scope	General description
Tier 5	Maximum scope	Endpoint in a Region farther across a large geographical area or different continents

Note

Customers can also use one free, local 500 Mbps interconnect per Region starting in May.

A higher Tier automatically includes all paths belonging to a lower Tier. Your Interconnect will be automatically subscribed to the lowest Tier that includes all the specific paths between the Regions where your workloads reside and your Interconnect. The same mechanism applies in the case of Cloud WAN when you add a new CNE to your Core Network. Examples are provided below to show how Tiers operate.

Tier operation with Global and Regional/Local use cases

By using a DXGW, you can use your Interconnect with supported AWS networking services: Virtual private gateways, Transit Gateways, and Cloud WAN.

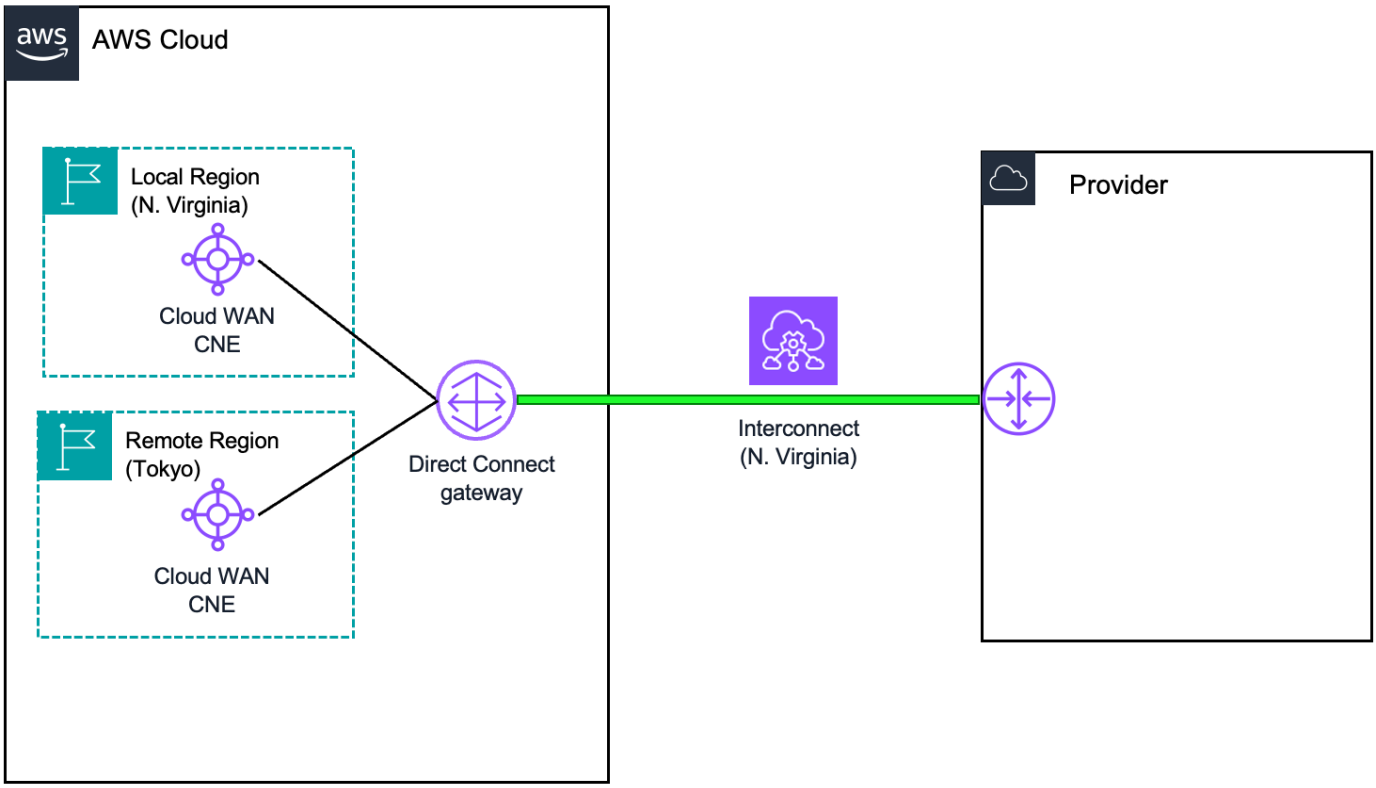
Global connectivity to Cloud WAN

Cloud WAN is a global networking service that spans across all AWS Regions where you have configured a Core Network Edge (CNE) to be a part of your Core Network. Using the native Direct Connect attachment, any Cloud WAN CNE can reach any Interconnect globally that is attached to the same DXGW.

For example:

- You have a global Core Network with CNEs in the us-east-1 (N. Virginia) and ap-northeast-1/Asia Pacific (Tokyo) Regions.
- You create an Interconnect in us-east-1 and attach it to a DXGW.
- In this configuration, us-east-1 is the local Region for your Interconnect, and ap-northeast-1 is a remote Region.
- You then associate that DXGW to a Core Network segment that has both CNEs.

In this configuration, both CNEs can reach your Interconnect using the shortest available path and without routing traffic through a Region.



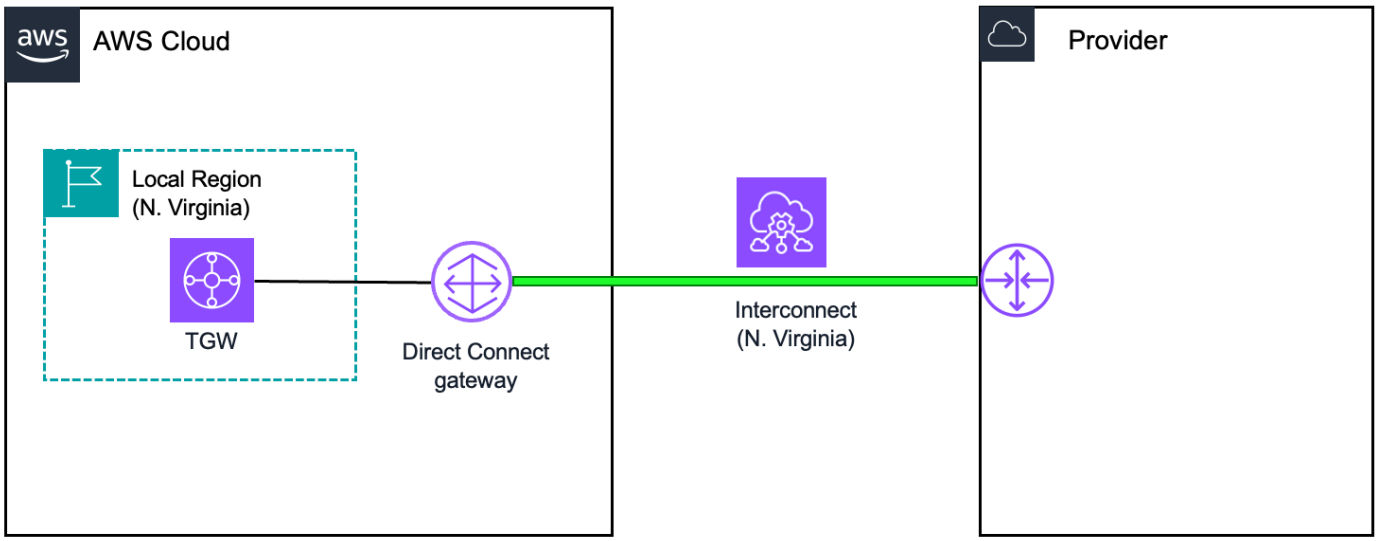
Local connectivity to Virtual gateways or Transit Gateways

Virtual gateways and Transit Gateways are regional networking services which reside in a single, specific AWS Region. When using Interconnect with regional networking services, a DXGW that has an Interconnect attachment will support associations to a VGW or TGW only in the AWS Region that is local for that Interconnect which you selected at its creation. If you attempt to attach an Interconnect to an existing DXGW that is already associated to a TGW in the local Region and also to a TGW in a remote Region, then the new attachment will fail.

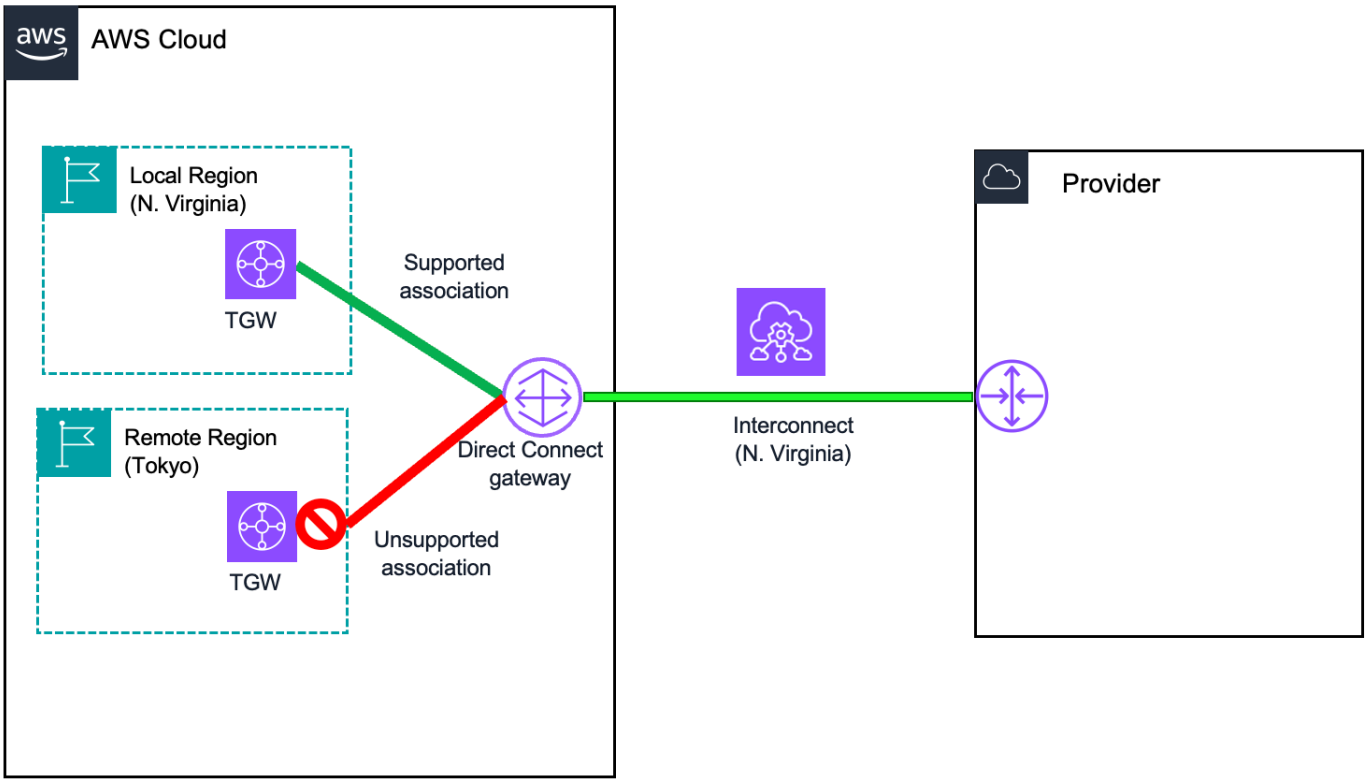
For example:

- You have a TGW in us-east-1 (N. Virginia) and another TGW in us-west-2 (Oregon).
- You created a 1 Gbps Interconnect and selected us-east-1 (N. Virginia) as the AWS Region.
- The DXGW that has that Interconnect attached will only be able to be associated to your TGW in us-east-1 (N. Virginia).

Valid local architecture:

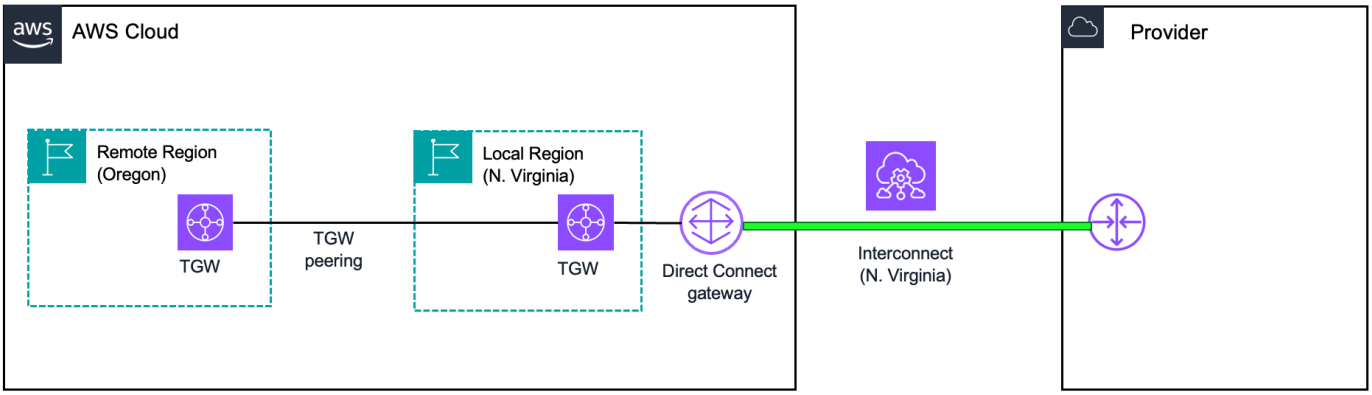


Unsupported architecture when using local networking services in a remote Region:



Note that architectures that use TGW peering across Regions are fully supported. Following the example above, your TGW in the remote Region, us-west-2 (Oregon), can reach your Interconnect in us-east-1 (N. Virginia) by peering to your TGW in that Region. Cross-region data transfer

charges and TGW data processing charges would apply. For more information, review the TGW documentation.

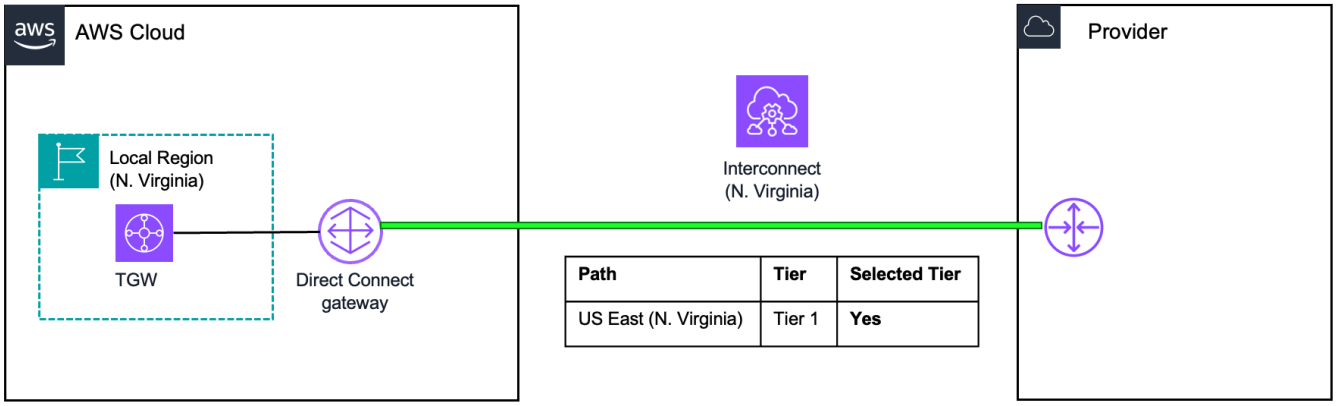


Example Scenarios

The following examples show how the Tier structure is applied to your Interconnect in different scenarios.

Example 1: Local, single Interconnect

You create a new Interconnect that is local to the us-east-1 (N. Virginia) Region and associate it to a VPC in the same Region using a VGW. This Interconnect will be subscribed to Tier 1 as the local path between the N. Virginia Region and an Interconnect provisioned in N. Virginia is Tier 1.

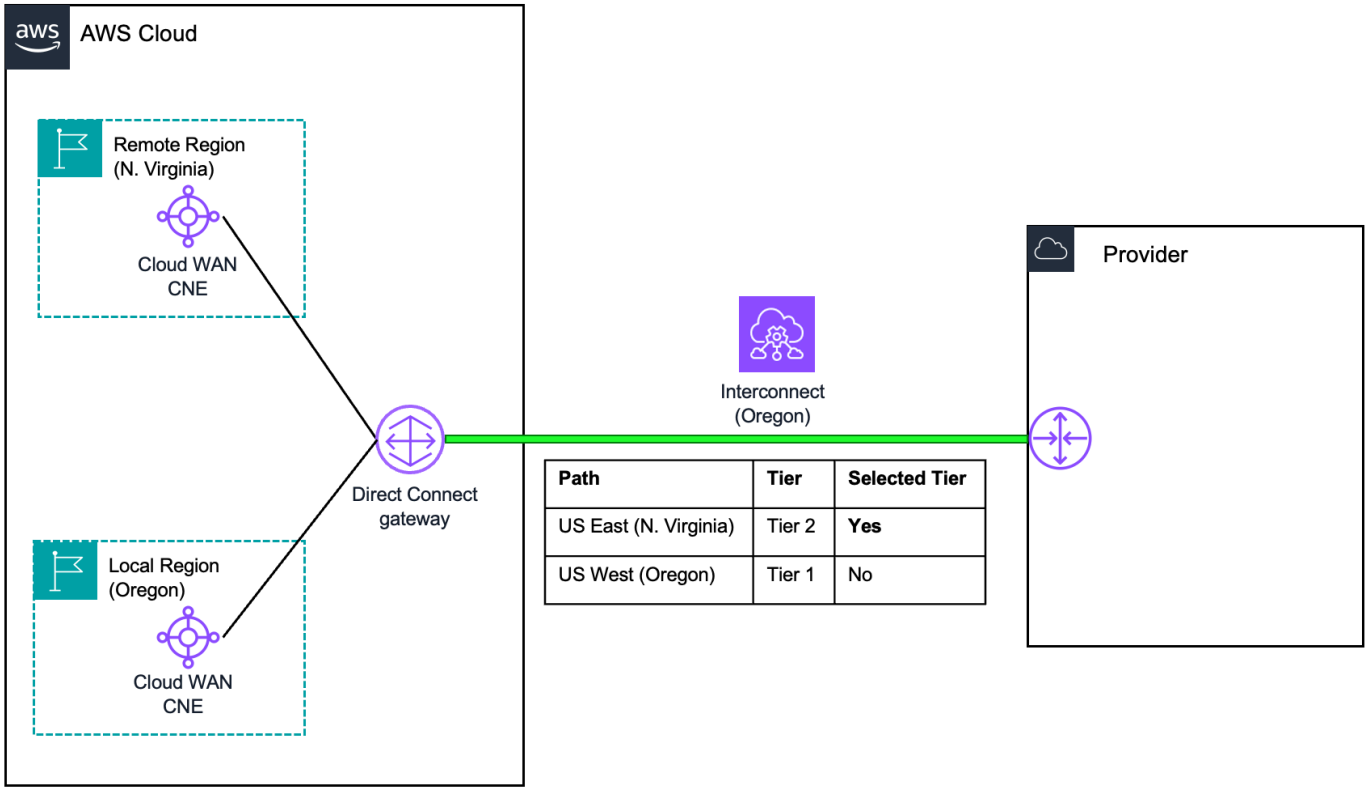


Path	Tier	Selected Tier
US East (N. Virginia)	Tier 1	**Yes**

Example 2: Global, same continent, single Interconnect

- You have a Cloud WAN Core Network with CNEs in the us-east-1 (N. Virginia) and us-west-2 (Oregon) Regions.
- You create a new Interconnect that is local to the Oregon Region and associate it to your Core Network.

In this case, the path between the Oregon Region and your Interconnect that is local to that Region is Tier 1 and the path between your Interconnect and the N. Virginia Region is Tier 2. This Interconnect will then be subscribed to Tier 2, as it is the lowest Tier that includes all your potential paths.

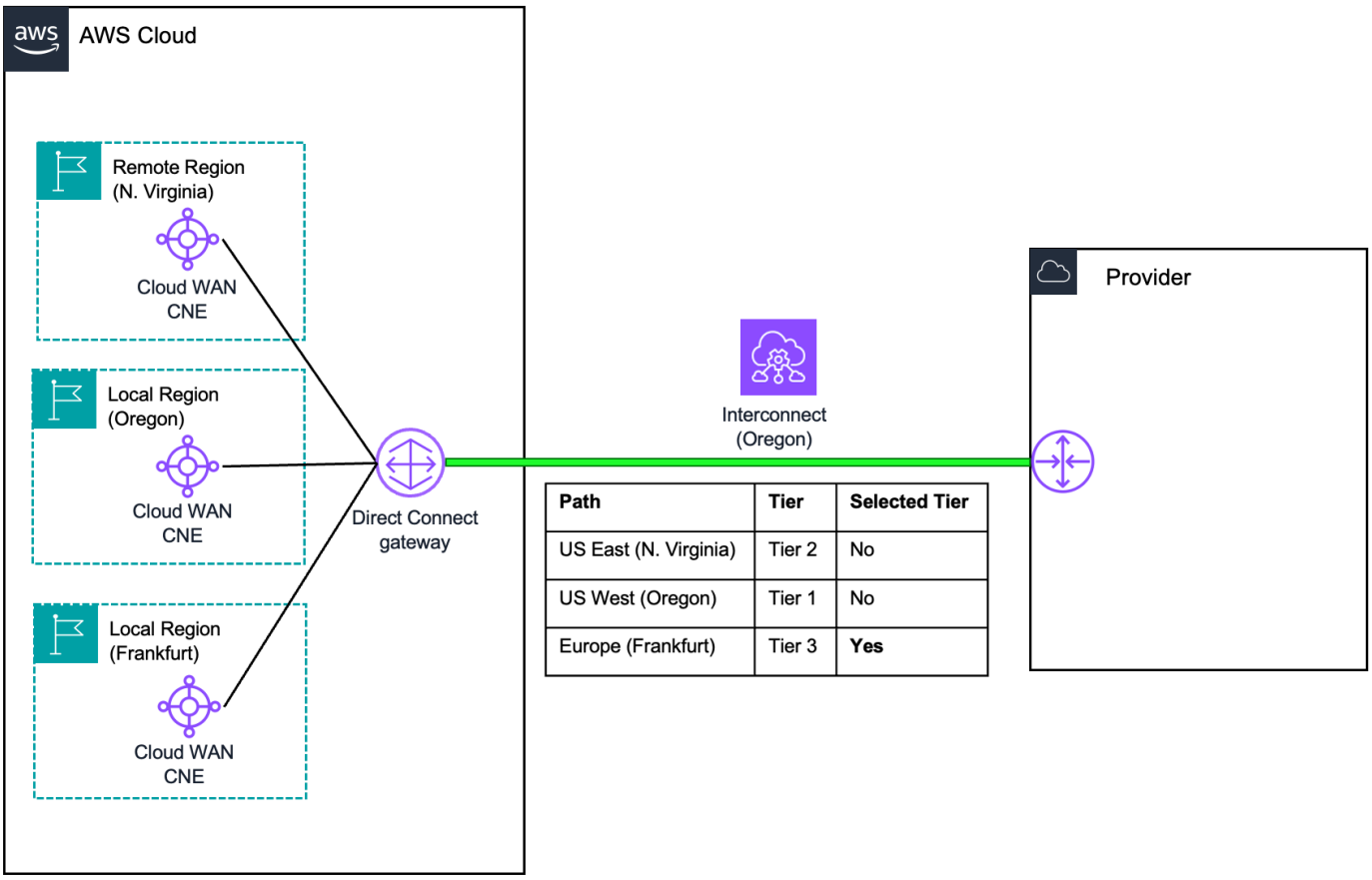


Path	Tier	Selected Tier
US East (N. Virginia)	Tier 2	**Yes**
US West (Oregon)	Tier 1	No

Example 3: Global, intercontinental, single Interconnect

- You have the same Cloud WAN Core Network with CNEs in the us-east-1 (N. Virginia) and us-west-2 (Oregon) Regions and an Interconnect that is local to the Oregon Region.
- You add a new CNE in the eu-central-1/Europe (Frankfurt) Region to the Core Network.

In this case, the new path between the Interconnect and the Frankfurt Region is Tier 3. The Interconnect will then be automatically upgraded to Tier 3, as it is the lowest Tier that now includes the local Tier 1 path between the Interconnect and the Oregon Region, the Tier 2 path between the Interconnect and the N. Virginia Region, and the Tier 3 path between the Interconnect and the Frankfurt Region.

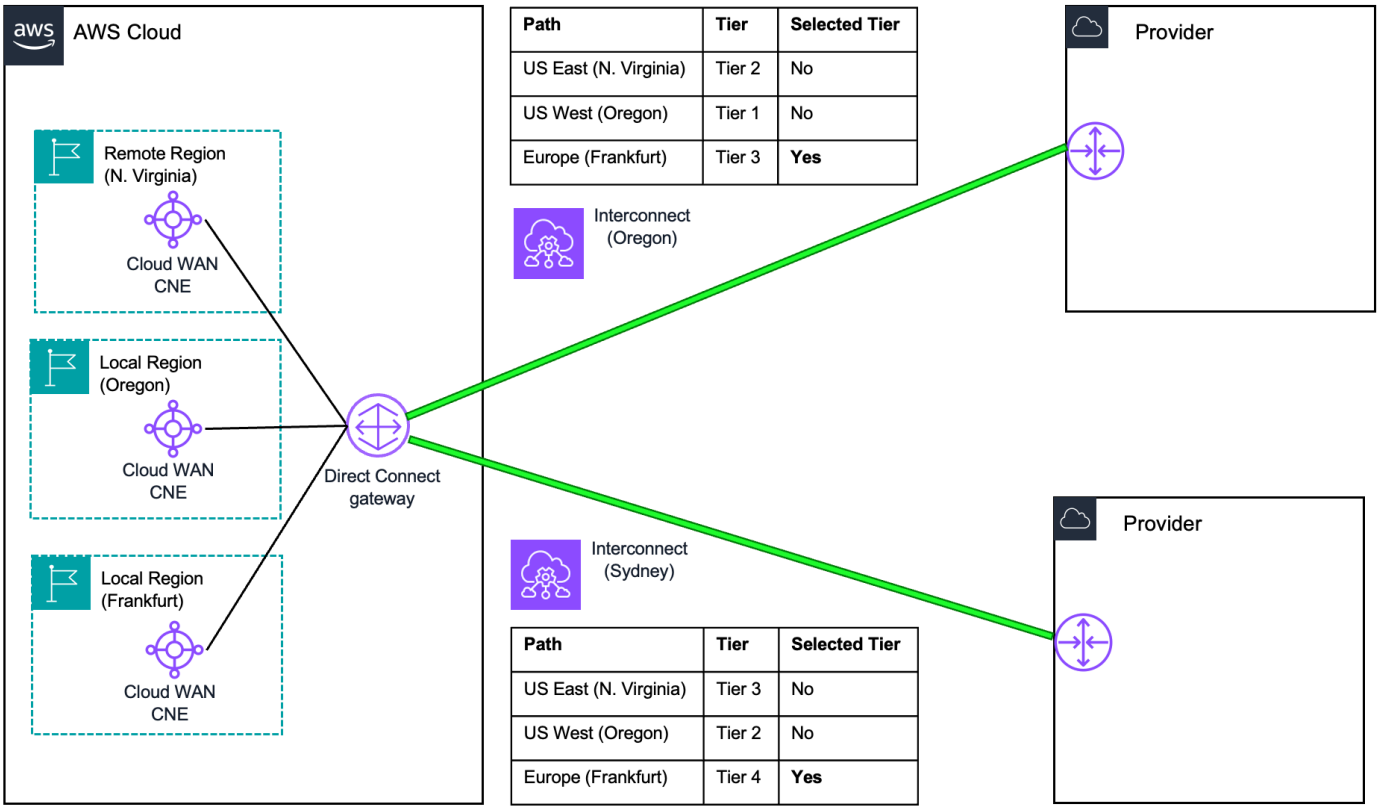


Example 4: Global, intercontinental, multiple Interconnects

- You have the same Cloud WAN Core Network with CNEs in the us-east-1 (N. Virginia), us-west-2 (Oregon), and eu-central-1/Europe (Frankfurt) Regions with an existing Interconnect that is local to the Oregon Region which is already subscribed to Tier 3.
- You don't have a Cloud WAN CNE in the Sydney Region, but you need to reach your resources in another CSP's Region in Sydney from your existing CNEs in Germany and the United States.
- You create a new Interconnect that is local to the ap-southeast-2/Asia Pacific (Sydney) Region and attach it to the same DXGW that is already associated to your Cloud WAN Core Network.

In this case, there is no change in Tier to your existing Interconnect as it doesn't need to reach a CNE in Sydney.

Your new Interconnect provisioned in Sydney will be automatically subscribed to Tier 4 as it needs to reach your CNEs in Germany and the United States and that is the lowest Tier that now includes the Tier 3 path between the Interconnect in Sydney and the Oregon Region, the Tier 2 path between Sydney and the N. Virginia Region, and the Tier 4 path between Sydney and the Frankfurt Region. You now have two Interconnects subscribed to Tiers 3 and 4, respectively, as those are the lowest possible Tiers that include all the possible paths for the specific Interconnect.



Path	Tier	Selected Tier
US East (N. Virginia)	Tier 2	No
US West (Oregon)	Tier 1	No
Europe (Frankfurt)	Tier 3	**Yes**

Path	Tier	Selected Tier
US East (N. Virginia)	Tier 3	No
US West (Oregon)	Tier 2	No
Europe (Frankfurt)	Tier 4	**Yes**

Security in AWS Interconnect

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. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to AWS Interconnect, 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 company's requirements, and applicable laws and regulations

This documentation helps you understand how to apply the shared responsibility model when using AWS Interconnect. It shows you how to configure AWS Interconnect to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your AWS Interconnect resources.

Contents

Identity and access management for AWS Interconnect

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 AWS Interconnect resources. IAM is an AWS service that you can use with no additional charge.

Topics

- Authenticating with identities
- Managing access using policies
- How AWS Interconnect works with IAM
- AWS Interconnect identity-based policy examples
- Troubleshooting AWS Interconnect identity and access

Audience

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

Service user – If you use the AWS Interconnect service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more AWS Interconnect 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 AWS Interconnect, see Troubleshooting AWS Interconnect identity and access.

Service administrator – If you're in charge of AWS Interconnect resources at your company, you probably have full access to AWS Interconnect. It's your job to determine which AWS Interconnect features and resources your service users 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 AWS Interconnect, see How AWS Interconnect works with IAM.

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

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. 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 sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users, your company's single sign-on

authentication, and your Google or Facebook credentials are examples of federated identities. When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, 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 Multi-factor authentication in the *AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide* and 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 don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the *Account Management Reference Guide*.

Federated identity

As a best practice, require human users, including users that require administrator access, to use federation with an identity provider to access AWS services by using temporary credentials.

A federated identity is a user from your enterprise user directory, a web identity provider, the AWS Directory Service, the Identity Center directory, or any user that accesses AWS services by using credentials provided through an identity source. When federated identities access AWS accounts, they assume roles, and the roles provide temporary credentials.

For centralized access management, we recommend that you use AWS IAM Identity Center. You can create users and groups in IAM Identity Center, or you can connect and synchronize to a set of users and groups in your own identity source for use across all your AWS accounts and applications. For information about IAM Identity Center, see What is IAM Identity Center? in the *AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide*.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see Rotate access keys regularly for use cases that require long-term credentials in the *IAM User Guide*.

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:

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see Creating a role for a third-party Identity Provider in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control

what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permissions sets, see Permission sets in the *AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide*.

- **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.

To learn whether to use IAM roles, 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 AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. 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. 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, role, or group. 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*.

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 Interconnect works with IAM

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

Topics

- AWS Interconnect Identity-based policies
- Authorization based on AWS Interconnect tags
- AWS Interconnect IAM roles

AWS Interconnect Identity-based policies

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. You can't specify the principal in an identity-based policy because it applies to the user or role to which it is attached. 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

The `Action` element of an IAM identity-based policy describes the specific action or actions that will be allowed or denied by the policy. Policy actions usually have the same name as the associated AWS API operation. The action is used in a policy to grant permissions to perform the associated operation.

Policy actions in AWS Interconnect use the following prefix before the action: `interconnect:`. For example, to grant someone permission to run an Amazon EC2 instance with the Amazon EC2 `RunInstances` API operation, you include the `ec2:RunInstances` action in their policy. Policy statements must include either an `Action` or `NotAction` element. AWS Interconnect 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": [
    "interconnect:CreateConnection",
    "interconnect:ListEnvironments"
]
```

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

```
"Action": "interconnect:Create*"
```

To see a list of AWS Interconnect actions, see Actions Defined by AWS Interconnect in the *IAM User Guide*.

Resources

The `Resource` element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. You specify a resource using an ARN or using the wildcard (*) to indicate that the statement applies to all resources.

The AWS Interconnect Connection resource has the following ARN:

```
arn:${Partition}:interconnect:${Region}:${Account}:connection/${ConnectionId}
```

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

For example, to specify the `mcc-12345678` connection in your statement, use the following ARN:

```
"Resource": "arn:aws:interconnect:us-east-1:123456789012:connection/mcc-12345678"
```

To specify all instances that belong to a specific account, use the wildcard (*):

```
"Resource": "arn:aws:interconnect:us-east-1:123456789012:connection/*"
```

Some AWS Interconnect actions, such as those for creating resources, cannot be performed on a specific resource. In those cases, you must use the wildcard (*).

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

To see a list of AWS Interconnect resource types and their ARNs, see Resources Defined by AWS Interconnect in the *IAM User Guide*. To learn with which actions you can specify the ARN of each resource, see Actions Defined by AWS Interconnect.

Condition keys

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 Interconnect defines its own set of condition keys and also supports using some global condition keys. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

All Amazon EC2 actions support the `aws:RequestedRegion` and `ec2:Region` condition keys. For more information, see Example: Restricting access to a specific region.

To see a list of AWS Interconnect condition keys, see Condition Keys for AWS Interconnect in the *IAM User Guide*. To learn with which actions and resources you can use a condition key, see Actions defined by AWS Interconnect.

Examples

To view examples of AWS Interconnect identity-based policies, see AWS Interconnect identity-based policy examples.

Authorization based on AWS Interconnect tags

You can attach tags to AWS Interconnect resources or pass tags in a request to AWS Interconnect. To control access based on tags, you provide tag information in the condition element of a policy using the `interconnect:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys. For more information about tagging AWS Interconnect resources, see AWS Interconnect TagResource API.

To view an example identity-based policy for limiting access to a resource based on the tags on that resource, see Viewing AWS Interconnect connections based on tags.

AWS Interconnect IAM roles

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

Using temporary credentials with AWS Interconnect

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.

AWS Interconnect supports using temporary credentials.

AWS Interconnect identity-based policy examples

By default, IAM users and roles don't have permission to create or modify AWS Interconnect 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

- Using the AWS Interconnect console
- Allow users to view their own permissions
- Viewing AWS Interconnect Connections based on tags

Policy best practices

Identity-based policies determine whether someone can create, access, or delete AWS Interconnect 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 with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the AWS managed policies that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.
- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.
- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as CloudFormation. For more information, see IAM JSON policy elements: condition in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see IAM Access Analyzer policy validation in the *IAM User Guide*.
- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or root users in your account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Configuring MFA-protected API access in the *IAM User Guide*.

Using the AWS Interconnect console

To access the AWS Interconnect console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the AWS Interconnect 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 (users or roles) with that policy.

To ensure that those entities can still use the AWS Interconnect console, also attach the following AWS managed policy to the entities. For more information, see Adding permissions to a user in the *IAM User Guide*:

```
AWSInterconnectConsoleAccess
```

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 AWS Interconnect Connections based on tags

You can use conditions in your identity-based policy to control access to AWS Interconnect resources based on tags. This example shows how you might create a policy that allows viewing a Connection. However, permission is granted only if the Connection 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": "ListConnectionsInConsole",
            "Effect": "Allow",
            "Action": "interconnect:ListConnections",
            "Resource": "*"
        },
        {
            "Sid": "ViewConnectionIfOwner",
            "Effect": "Allow",
            "Action": "interconnect:GetConnection",
            "Resource": "arn:aws:interconnect:*:*:connection/*",
            "Condition": {
                "StringEquals": {"interconnect: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 an AWS Interconnect Connection, the Connection must be tagged `Owner=richard-roe` or `owner=richard-roe`. Otherwise he is denied 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*.

Troubleshooting AWS Interconnect identity and access

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

Topics

- I am not authorized to perform an action in AWS Interconnect
- I am not authorized to perform iam:PassRole
- I want to allow people outside of my AWS account to access my AWS Interconnect resources

I am not authorized to perform an action in AWS Interconnect

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 Connection but does not have `interconnect:GetConnection` permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 interconnect:GetConnection on resource: mcc-12345678
```

In this case, Mateo asks his administrator to update his policies to allow him to access the `mcc-12345678` resource using the `interconnect:GetConnection` 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 AWS Interconnect.

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 AWS Interconnect. 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 allow people outside of my AWS account to access my AWS Interconnect 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 AWS Interconnect supports these features, see How AWS Interconnect works with IAM.
- 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*.

Infrastructure security in AWS Interconnect

Service Protections and Access

As a managed service, AWS Interconnect is protected by the AWS global network security procedures that are described in the Amazon Web Services: Overview of Security Processes whitepaper.

You use AWS published API calls to access AWS Interconnect 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 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.

Data-in-Transit Protections

The managed connections that you obtain through AWS Interconnect traverse between the AWS Network and the respective partner's network across MACsec encrypted links. The traffic you send secured in transit through the AWS Network, and the corresponding link to the partner's network.

> **Note**
>
> Please consult the documentation provided by your respective partner to determine the level of data transit security promised on the partner's network.

Monitoring AWS Interconnect

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS Interconnect and your other AWS solutions. AWS provides the following monitoring tools to watch AWS Interconnect, report when something is wrong, and take automatic actions when appropriate:

- *Amazon CloudWatch* monitors your AWS resources and and the applications you run on AWS in 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.
- *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.

Contents

Monitoring AWS Interconnect API calls using AWS CloudWatch

Learn how AWS CloudWatch can be used to monitoring AWS Interconnect resources.

== Monitoring AWS Interconnect with Amazon CloudWatch

You can monitor AWS Interconnect using CloudWatch, which collects raw data and processes it into readable, near real-time metrics. These statistics are kept for 15 months, so that you can access historical information and gain a better 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.

For AWS Interconnect, once you have established workloads that are generating consistent levels of traffic, you may want to watch and monitor the `ConnectionBpsEgress` and `ConnectionBpsIngress` metrics to confirm expected levels of utilization.

The following tables list the metrics and dimensions for AWS Interconnect.

Metric	Description
ConnectionBpsEgress	The bitrate for outbound data from the AWS side of the connection. The number reported is the aggregate (average) over the specified time period (5 minutes by default, 1 minute minimum). You can change the default aggregate. This metric might be unavailable for a new connection, or when a device reboots. The metric starts when the connection is used to send or receive traffic. Units: Bits per second
ConnectionBpsIngress	The bitrate for inbound data to the AWS side of the connection. This metric might be unavailable for a new connection, or when a device reboots. The metric starts when the connection is used to send or receive traffic. Units: Bits per second
ConnectionPpsEgress	The packet rate for outbound data from the AWS side of the connection. The number reported is the aggregate (average) over the specified time period (5 minutes by default, 1 minute minimum). You can change the default aggregate. This metric might be unavailable for a new connection, or when a device reboots. The metric starts when the connection is used to send or receive traffic. Units: Packets per second

Metric	Description
ConnectionPpsIngress	The packet rate for inbound data to the AWS side of the connection. The number reported is the aggregate (average) over the specified time period (5 minutes by default, 1 minute minimum). You can change the default aggregate. This metric might be unavailable for a new connection, or when a device reboots. The metric starts when the connection is used to send or receive traffic. Units: Packets per second

Logging AWS Interconnect API calls using AWS CloudTrail

AWS Interconnect is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in AWS Interconnect. CloudTrail captures all API calls for AWS Interconnect as events. The calls captured include calls from the AWS Interconnect console and code calls to the AWS Interconnect API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AWS Interconnect. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to AWS Interconnect, the IP address from which the request was made, who made the request, when it was made, and additional details.

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

AWS Interconnect information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS Interconnect, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for AWS Interconnect, 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
- Receiving CloudTrail log files from multiple accounts

All AWS Interconnect actions are logged by CloudTrail and are documented in the AWS Interconnect API Reference. For example, calls to the `CreateConnection`, `AcceptConnectionPropsoal` and `DeleteConnection` actions 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.
- Whether the request was made on the partner side of the service.

For more information, see the CloudTrail userIdentity element.

Understanding AWS Interconnect log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the `CreateConnection` action.

```
{
    "eventVersion": "1.11",
    "userIdentity": {
        ... redacted ...
    },
    "eventTime": "2026-04-12T21:12:05Z",
    "eventSource": "interconnect.amazonaws.com",
    "eventName": "CreateConnection",
    "awsRegion": "eu-west-2",
    "sourceIPAddress": "174.179.26.172",
    "userAgent": "Mozilla/5.0 ...",
    "requestParameters": {
        "description": "My Multicloud Connection to GCP",
        "bandwidth": "1Gbps",
        "attachPoint": {
            "directConnectGateway": "ba72fdf5-e244-45ac-8374-cf9ebddd4d90"
        },
        "environmentId": "mce-aws-gcp-lhr",
        "remoteAccount": {
            "identifier": "123412341234"
        },
        "tags": {},
        "clientToken": "dcc6ca87-6770-4765-b1e6-5504b9a61bdc"
    },
    "responseElements": {
        "connection": {
            "id": "mcc-12345678",
            "arn": "arn:aws:interconnect:eu-west-2:000000000000:connection/
mcc-12345678",
            "description": "Test",
            "bandwidth": "1Gbps",
            "attachPoint": {
                "directConnectGateway": "ba72fdf5-e244-45ac-8374-cf9ebddd4d90"
            },
            "environmentId": "mce-nullprov",
            "provider": {
                "cloudServiceProvider": "aws-test"
            },
            "location": "eu-west-1",
            "type": "Multicloud",
            "state": "requested",
            "sharedId": "ccc4c1c4-cb96-468a-cac9-fc8118a309db",
            "ownerAccount": "000000000000",
```

```
            "activationKey": "HIDDEN_DUE_TO_SECURITY_REASONS",
            "tags": {}
        }
    },
    "requestID": "7e60af38-196d-477f-85a0-114efe569fe8",
    "eventID": "5bc65547-bfcf-47c9-a541-1dc87acbd717",
    "readOnly": false,
    "resources": [
        {
            "accountId": "000000000000",
            "type": "AWS::INTERCONNECT::Connection",
            "ARN": "arn:aws:interconnect:eu-west-2:000000000000:connection/
mcc-12345678"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "000000000000",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "interconnect.eu-west-2.api.aws"
    },
    "sessionCredentialFromConsole": "true"
}
```

Creating AWS Interconnect resources with AWS CloudFormation

AWS Interconnect is integrated with AWS CloudFormation, a service that helps you model and set up your AWS resources so that you can spend less time creating and managing your resources and infrastructure. You create a template that describes all the AWS resources that you want (like Connections), and AWS CloudFormation takes care of provisioning and configuring those resources for you.

When you use AWS CloudFormation, you can reuse your template to set up your AWS Interconnect resources consistently and repeatedly. Just describe your resources once, and then provision the same resources over and over in multiple AWS accounts and Regions.

AWS Interconnect and AWS CloudFormation templates

To provision and configure resources for AWS Interconnect and related services, you must understand AWS CloudFormation templates. Templates are formatted text files in JSON or YAML. These templates describe the resources that you want to provision in your AWS CloudFormation stacks. If you're unfamiliar with JSON or YAML, you can use AWS CloudFormation Designer to help you get started with AWS CloudFormation templates. For more information, see What is AWS CloudFormation Designer? in the *AWS CloudFormation User Guide*.

AWS Interconnect supports creating Connections in AWS CloudFormation. For more information, including examples of JSON and YAML templates for Connections, see AWS Interconnect Connection in the *AWS CloudFormation User Guide*.

Learn more about AWS CloudFormation

To learn more about AWS CloudFormation, see the following resources:

- AWS CloudFormation
- AWS CloudFormation User Guide
- AWS CloudFormation Command Line Interface User Guide

Quotas for AWS Interconnect

Quotas for AWS Interconnect

Your AWS account has default quotas, formerly referred to as limits, for each AWS service. Unless otherwise noted, each quota is Region-specific. You can request increases for some quotas, and other quotas cannot be increased.

To view the quotas for AWS Interconnect, open the Service Quotas console. In the navigation pane, choose **AWS services** and select **AWS Interconnect**.

To request a quota increase, see Requesting a Quota Increase in the *Service Quotas User Guide*. If the quota is not yet available in Service Quotas, use the limit increase form.

Your AWS account has the following quotas related to AWS Interconnect.

Component	Quota	Description
Interconnect Maximum Created Connections	10	The maximum number of AWS Interconnect Connections allowed per account.
Interconnect Outstanding Requested Connections	4	The maximum number of AWS Interconnect Connections in the `requested` state allowed per account.
Multicloud Connections Per Provider	2	The maximum number of AWS Interconnect multicloud Connections allowed per provider per account.

Component	Quota	Description
Last Mile Connections Per Provider	2	The maximum number of AWS Interconnect last mile Connections allowed per provider per account.