AWS Direct Connect

API Reference

API Version 2012-10-25

AWS Direct Connect: API Reference

Table of Contents

Welcome

AWS Direct Connect links your internal network to an AWS Direct Connect location over a standard Ethernet fiber-optic cable. One end of the cable is connected to your router, the other to an AWS Direct Connect router. With this connection in place, you can create virtual interfaces directly to the AWS cloud (for example, to Amazon EC2 and Amazon S3) and to Amazon VPC, bypassing Internet service providers in your network path. A connection provides access to all AWS Regions except the China (Beijing) and (China) Ningxia Regions. AWS resources in the China Regions can only be accessed through locations associated with those Regions.

This document was last published on November 19, 2018.

Actions

The following actions are supported:

AllocateConnectionOnInterconnect

Deprecated. Use AllocateHostedConnection (p. 8) instead.

Creates a hosted connection on an interconnect.

Allocates a VLAN number and a specified amount of bandwidth for use by a hosted connection on the specified interconnect.

Note
Intended for use by AWS Direct Connect partners only.

Request Syntax

```
{
   "bandwidth": "string",
   "connectionName": "string",
   "interconnectId": "string",
   "ownerAccount": "string",
   "vlan": number
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

bandwidth (p. 4)

The bandwidth of the connection, in Mbps. The possible values are 50Mbps, 100Mbps, 200Mbps, 300Mbps, 400Mbps, and 500Mbps.

Type: String

Required: Yes

connectionName (p. 4)

The name of the provisioned connection.

Type: String

Required: Yes

interconnectId (p. 4)

The ID of the interconnect on which the connection will be provisioned. For example, dxcon-456abc78.

Type: String

Required: Yes

ownerAccount (p. 4)

The ID of the AWS account of the customer for whom the connection will be provisioned.

Type: String

Required: Yes

vlan (p. 4)

The dedicated VLAN provisioned to the connection.

Type: Integer

Required: Yes

Response Syntax

```
{
   "awsDevice": "string",
   "awsDeviceV2": "string",
   "bandwidth": "string",
   "connectionId": "string",
   "connectionName": "string",
   "connectionState": "string",
   "hasLogicalRedundancy": "string",
   "jumboFrameCapable": boolean,
   "lagId": "string",
   "loaIssueTime": number,
   "location": "string",
   "ownerAccount": "string",
   "partnerName": "string",
   "region": "string",
   "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

awsDevice (p. 5)

This parameter has been deprecated.

The Direct Connect endpoint on which the physical connection terminates.

Type: String

awsDeviceV2 (p. 5)

The Direct Connect endpoint on which the physical connection terminates.

Type: String

bandwidth (p. 5)

The bandwidth of the connection.

Type: String

connectionId (p. 5)

The ID of the connection.

Type: String

connectionName (p. 5)

The name of the connection.

Type: String

connectionState (p. 5)

The state of the connection. The following are the possible values:

- `ordering`: The initial state of a hosted connection provisioned on an interconnect. The connection stays in the ordering state until the owner of the hosted connection confirms or declines the connection order.
- `requested`: The initial state of a standard connection. The connection stays in the requested state until the Letter of Authorization (LOA) is sent to the customer.
- `pending`: The connection has been approved and is being initialized.
- `available`: The network link is up and the connection is ready for use.
- `down`: The network link is down.
- `deleting`: The connection is being deleted.
- `deleted`: The connection has been deleted.
- `rejected`: A hosted connection in the `ordering` state enters the `rejected` state if it is deleted by the customer.

Type: String

Valid Values: `ordering | requested | pending | available | down | deleting | deleted | rejected`

hasLogicalRedundancy (p. 5)

Indicates whether the connection supports a secondary BGP peer in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

jumboFrameCapable (p. 5)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

lagId (p. 5)

The ID of the LAG.

Type: String

loaIssueTime (p. 5)

The time of the most recent call to DescribeLoa (p. 113) for this connection.

Type: Timestamp

location (p. 5)

The location of the connection.

Type: String

ownerAccount (p. 5)

The ID of the AWS account that owns the connection.

Type: String

partnerName (p. 5)

The name of the AWS Direct Connect service provider associated with the connection.

Type: String

region (p. 5)

The AWS Region where the connection is located.

Type: String

vlan (p. 5)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

AllocateHostedConnection

Creates a hosted connection on the specified interconnect or a link aggregation group (LAG).

Allocates a VLAN number and a specified amount of bandwidth for use by a hosted connection on the specified interconnect or LAG.

Note
Intended for use by AWS Direct Connect partners only.

Request Syntax

```
{
   "bandwidth": "string",
   "connectionId": "string",
   "connectionName": "string",
   "ownerAccount": "string",
   "vlan": number
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

bandwidth (p. 8)

The bandwidth of the hosted connection, in Mbps. The possible values are 50Mbps, 100Mbps, 200Mbps, 300Mbps, 400Mbps, and 500Mbps.

Type: String

Required: Yes

connectionId (p. 8)

The ID of the interconnect or LAG.

Type: String

Required: Yes

connectionName (p. 8)

The name of the hosted connection.

Type: String

Required: Yes

ownerAccount (p. 8)

The ID of the AWS account ID of the customer for the connection.

Type: String

Required: Yes

vlan (p. 8)

The dedicated VLAN provisioned to the hosted connection.

Type: Integer

Required: Yes

Response Syntax

```
{
   "awsDevice": "string",
   "awsDeviceV2": "string",
   "bandwidth": "string",
   "connectionId": "string",
   "connectionName": "string",
   "connectionState": "string",
   "hasLogicalRedundancy": "string",
   "jumboFrameCapable": boolean,
   "lagId": "string",
   "loaIssueTime": number,
   "location": "string",
   "ownerAccount": "string",
   "partnerName": "string",
   "region": "string",
   "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

awsDevice (p. 9)

This parameter has been deprecated.

The Direct Connect endpoint on which the physical connection terminates.

Type: String

awsDeviceV2 (p. 9)

The Direct Connect endpoint on which the physical connection terminates.

Type: String

bandwidth (p. 9)

The bandwidth of the connection.

Type: String

connectionId (p. 9)

The ID of the connection.

Type: String

connectionName (p. 9)

The name of the connection.

Type: String

connectionState (p. 9)

The state of the connection. The following are the possible values:

- `ordering`: The initial state of a hosted connection provisioned on an interconnect. The connection stays in the ordering state until the owner of the hosted connection confirms or declines the connection order.
- `requested`: The initial state of a standard connection. The connection stays in the requested state until the Letter of Authorization (LOA) is sent to the customer.
- `pending`: The connection has been approved and is being initialized.
- `available`: The network link is up and the connection is ready for use.
- `down`: The network link is down.
- `deleting`: The connection is being deleted.
- `deleted`: The connection has been deleted.
- `rejected`: A hosted connection in the `ordering` state enters the `rejected` state if it is deleted by the customer.

Type: String

Valid Values: `ordering | requested | pending | available | down | deleting | deleted | rejected`

hasLogicalRedundancy (p. 9)

Indicates whether the connection supports a secondary BGP peer in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

jumboFrameCapable (p. 9)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

lagId (p. 9)

The ID of the LAG.

Type: String

loaIssueTime (p. 9)

The time of the most recent call to DescribeLoa (p. 113) for this connection.

Type: Timestamp

location (p. 9)

The location of the connection.

Type: String

ownerAccount (p. 9)

The ID of the AWS account that owns the connection.

Type: String

partnerName (p. 9)

The name of the AWS Direct Connect service provider associated with the connection.

Type: String

region (p. 9)

The AWS Region where the connection is located.

Type: String

vlan (p. 9)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

AllocatePrivateVirtualInterface

Provisions a private virtual interface to be owned by the specified AWS account.

Virtual interfaces created using this action must be confirmed by the owner using ConfirmPrivateVirtualInterface (p. 37). Until then, the virtual interface is in the `Confirming` state and is not available to handle traffic.

Request Syntax

```
{
   "connectionId": "string",
   "newPrivateVirtualInterfaceAllocation": {
      "addressFamily": "string",
      "amazonAddress": "string",
      "asn": number,
      "authKey": "string",
      "customerAddress": "string",
      "mtu": number,
      "virtualInterfaceName": "string",
      "vlan": number
   },
   "ownerAccount": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 12)

The ID of the connection on which the private virtual interface is provisioned.

Type: String

Required: Yes

newPrivateVirtualInterfaceAllocation (p. 12)

Information about the private virtual interface.

Type: NewPrivateVirtualInterfaceAllocation (p. 163) object

Required: Yes

ownerAccount (p. 12)

The ID of the AWS account that owns the virtual private interface.

Type: String

Required: Yes

Response Syntax

```
{
```

```
   "addressFamily": "string",
   "amazonAddress": "string",
   "amazonSideAsn": number,
   "asn": number,
   "authKey": "string",
   "awsDeviceV2": "string",
   "bgpPeers": [
      {
         "addressFamily": "string",
         "amazonAddress": "string",
         "asn": number,
         "authKey": "string",
         "awsDeviceV2": "string",
         "bgpPeerId": "string",
         "bgpPeerState": "string",
         "bgpStatus": "string",
         "customerAddress": "string"
      }
   ],
   "connectionId": "string",
   "customerAddress": "string",
   "customerRouterConfig": "string",
   "directConnectGatewayId": "string",
   "jumboFrameCapable": boolean,
   "location": "string",
   "mtu": number,
   "ownerAccount": "string",
   "region": "string",
   "routeFilterPrefixes": [
      {
         "cidr": "string"
      }
   ],
   "virtualGatewayId": "string",
   "virtualInterfaceId": "string",
   "virtualInterfaceName": "string",
   "virtualInterfaceState": "string",
   "virtualInterfaceType": "string",
   "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

addressFamily (p. 12)

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

amazonAddress (p. 12)

The IP address assigned to the Amazon interface.

Type: String

amazonSideAsn (p. 12)

The autonomous system number (ASN) for the Amazon side of the connection.

Type: Long

asn (p. 12)

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

authKey (p. 12)

The authentication key for BGP configuration.

Type: String

awsDeviceV2 (p. 12)

The Direct Connect endpoint on which the virtual interface terminates.

Type: String

bgpPeers (p. 12)

The BGP peers configured on this virtual interface.

Type: Array of BGPPeer (p. 142) objects

connectionId (p. 12)

The ID of the connection.

Type: String

customerAddress (p. 12)

The IP address assigned to the customer interface.

Type: String

customerRouterConfig (p. 12)

The customer router configuration.

Type: String

directConnectGatewayId (p. 12)

The ID of the Direct Connect gateway.

Type: String

jumboFrameCapable (p. 12)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

location (p. 12)

The location of the connection.

Type: String

mtu (p. 12)

The maximum transmission unit (MTU), in bytes. The supported values are 1500 and 9001. The default value is 1500.

Type: Integer

ownerAccount (p. 12)

The ID of the AWS account that owns the virtual interface.

Type: String

region (p. 12)

The AWS Region where the virtual interface is located.

Type: String

routeFilterPrefixes (p. 12)

The routes to be advertised to the AWS network in this Region. Applies to public virtual interfaces.

Type: Array of RouteFilterPrefix (p. 170) objects

virtualGatewayId (p. 12)

The ID of the virtual private gateway. Applies only to private virtual interfaces.

Type: String

virtualInterfaceId (p. 12)

The ID of the virtual interface.

Type: String

virtualInterfaceName (p. 12)

The name of the virtual interface assigned by the customer network.

Type: String

virtualInterfaceState (p. 12)

The state of the virtual interface. The following are the possible values:

- `confirming`: The creation of the virtual interface is pending confirmation from the virtual interface owner. If the owner of the virtual interface is different from the owner of the connection on which it is provisioned, then the virtual interface will remain in this state until it is confirmed by the virtual interface owner.
- `verifying`: This state only applies to public virtual interfaces. Each public virtual interface needs validation before the virtual interface can be created.
- `pending`: A virtual interface is in this state from the time that it is created until the virtual interface is ready to forward traffic.
- `available`: A virtual interface that is able to forward traffic.
- `down`: A virtual interface that is BGP down.
- `deleting`: A virtual interface is in this state immediately after calling DeleteVirtualInterface (p. 89) until it can no longer forward traffic.
- `deleted`: A virtual interface that cannot forward traffic.
- `rejected`: The virtual interface owner has declined creation of the virtual interface. If a virtual interface in the `Confirming` state is deleted by the virtual interface owner, the virtual interface enters the `Rejected` state.

Type: String

Valid Values: `confirming | verifying | pending | available | down | deleting | deleted | rejected`

virtualInterfaceType (p. 12)

The type of virtual interface. The possible values are `private` and `public`.

Type: String

vlan (p. 12)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

AllocatePublicVirtualInterface

Provisions a public virtual interface to be owned by the specified AWS account.

The owner of a connection calls this function to provision a public virtual interface to be owned by the specified AWS account.

Virtual interfaces created using this function must be confirmed by the owner using ConfirmPublicVirtualInterface (p. 40). Until this step has been completed, the virtual interface is in the `confirming` state and is not available to handle traffic.

When creating an IPv6 public virtual interface, omit the Amazon address and customer address. IPv6 addresses are automatically assigned from the Amazon pool of IPv6 addresses; you cannot specify custom IPv6 addresses.

Request Syntax

```
{
   "connectionId": "string",
   "newPublicVirtualInterfaceAllocation": {
      "addressFamily": "string",
      "amazonAddress": "string",
      "asn": number,
      "authKey": "string",
      "customerAddress": "string",
      "routeFilterPrefixes": [
         {
            "cidr": "string"
         }
      ],
      "virtualInterfaceName": "string",
      "vlan": number
   },
   "ownerAccount": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 17)

The ID of the connection on which the public virtual interface is provisioned.

Type: String

Required: Yes

newPublicVirtualInterfaceAllocation (p. 17)

Information about the public virtual interface.

Type: NewPublicVirtualInterfaceAllocation (p. 167) object

Required: Yes

ownerAccount (p. 17)

The ID of the AWS account that owns the public virtual interface.

Type: String

Required: Yes

Response Syntax

```
{
   "addressFamily": "string",
   "amazonAddress": "string",
   "amazonSideAsn": number,
   "asn": number,
   "authKey": "string",
   "awsDeviceV2": "string",
   "bgpPeers": [
      {
         "addressFamily": "string",
         "amazonAddress": "string",
         "asn": number,
         "authKey": "string",
         "awsDeviceV2": "string",
         "bgpPeerId": "string",
         "bgpPeerState": "string",
         "bgpStatus": "string",
         "customerAddress": "string"
      }
   ],
   "connectionId": "string",
   "customerAddress": "string",
   "customerRouterConfig": "string",
   "directConnectGatewayId": "string",
   "jumboFrameCapable": boolean,
   "location": "string",
   "mtu": number,
   "ownerAccount": "string",
   "region": "string",
   "routeFilterPrefixes": [
      {
         "cidr": "string"
      }
   ],
   "virtualGatewayId": "string",
   "virtualInterfaceId": "string",
   "virtualInterfaceName": "string",
   "virtualInterfaceState": "string",
   "virtualInterfaceType": "string",
   "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

addressFamily (p. 18)

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

amazonAddress (p. 18)

The IP address assigned to the Amazon interface.

Type: String

amazonSideAsn (p. 18)

The autonomous system number (ASN) for the Amazon side of the connection.

Type: Long

asn (p. 18)

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

authKey (p. 18)

The authentication key for BGP configuration.

Type: String

awsDeviceV2 (p. 18)

The Direct Connect endpoint on which the virtual interface terminates.

Type: String

bgpPeers (p. 18)

The BGP peers configured on this virtual interface.

Type: Array of BGPPeer (p. 142) objects

connectionId (p. 18)

The ID of the connection.

Type: String

customerAddress (p. 18)

The IP address assigned to the customer interface.

Type: String

customerRouterConfig (p. 18)

The customer router configuration.

Type: String

directConnectGatewayId (p. 18)

The ID of the Direct Connect gateway.

Type: String

jumboFrameCapable (p. 18)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

location (p. 18)

The location of the connection.

Type: String

mtu (p. 18)

The maximum transmission unit (MTU), in bytes. The supported values are 1500 and 9001. The default value is 1500.

Type: Integer

ownerAccount (p. 18)

The ID of the AWS account that owns the virtual interface.

Type: String

region (p. 18)

The AWS Region where the virtual interface is located.

Type: String

routeFilterPrefixes (p. 18)

The routes to be advertised to the AWS network in this Region. Applies to public virtual interfaces.

Type: Array of RouteFilterPrefix (p. 170) objects

virtualGatewayId (p. 18)

The ID of the virtual private gateway. Applies only to private virtual interfaces.

Type: String

virtualInterfaceId (p. 18)

The ID of the virtual interface.

Type: String

virtualInterfaceName (p. 18)

The name of the virtual interface assigned by the customer network.

Type: String

virtualInterfaceState (p. 18)

The state of the virtual interface. The following are the possible values:

- `confirming`: The creation of the virtual interface is pending confirmation from the virtual interface owner. If the owner of the virtual interface is different from the owner of the connection on which it is provisioned, then the virtual interface will remain in this state until it is confirmed by the virtual interface owner.
- `verifying`: This state only applies to public virtual interfaces. Each public virtual interface needs validation before the virtual interface can be created.
- `pending`: A virtual interface is in this state from the time that it is created until the virtual interface is ready to forward traffic.
- `available`: A virtual interface that is able to forward traffic.
- `down`: A virtual interface that is BGP down.
- `deleting`: A virtual interface is in this state immediately after calling DeleteVirtualInterface (p. 89) until it can no longer forward traffic.

- `deleted`: A virtual interface that cannot forward traffic.
- `rejected`: The virtual interface owner has declined creation of the virtual interface. If a virtual interface in the `Confirming` state is deleted by the virtual interface owner, the virtual interface enters the `Rejected` state.

Type: String

Valid Values: `confirming | verifying | pending | available | down | deleting | deleted | rejected`

virtualInterfaceType (p. 18)

The type of virtual interface. The possible values are `private` and `public`.

Type: String

vlan (p. 18)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

AssociateConnectionWithLag

Associates an existing connection with a link aggregation group (LAG). The connection is interrupted and re-established as a member of the LAG (connectivity to AWS is interrupted). The connection must be hosted on the same AWS Direct Connect endpoint as the LAG, and its bandwidth must match the bandwidth for the LAG. You can re-associate a connection that's currently associated with a different LAG; however, if removing the connection would cause the original LAG to fall below its setting for minimum number of operational connections, the request fails.

Any virtual interfaces that are directly associated with the connection are automatically re-associated with the LAG. If the connection was originally associated with a different LAG, the virtual interfaces remain associated with the original LAG.

For interconnects, any hosted connections are automatically re-associated with the LAG. If the interconnect was originally associated with a different LAG, the hosted connections remain associated with the original LAG.

Request Syntax

```
{
   "connectionId": "string",
   "lagId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 22)

The ID of the connection. For example, dxcon-abc123.

Type: String

Required: Yes

lagId (p. 22)

The ID of the LAG with which to associate the connection. For example, dxlag-abc123.

Type: String

Required: Yes

Response Syntax

```
{
   "awsDevice": "string",
   "awsDeviceV2": "string",
   "bandwidth": "string",
   "connectionId": "string",
   "connectionName": "string",
```

```
   "connectionState": "string",
   "hasLogicalRedundancy": "string",
   "jumboFrameCapable": boolean,
   "lagId": "string",
   "loaIssueTime": number,
   "location": "string",
   "ownerAccount": "string",
   "partnerName": "string",
   "region": "string",
   "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

awsDevice (p. 22)

This parameter has been deprecated.

The Direct Connect endpoint on which the physical connection terminates.

Type: String

awsDeviceV2 (p. 22)

The Direct Connect endpoint on which the physical connection terminates.

Type: String

bandwidth (p. 22)

The bandwidth of the connection.

Type: String

connectionId (p. 22)

The ID of the connection.

Type: String

connectionName (p. 22)

The name of the connection.

Type: String

connectionState (p. 22)

The state of the connection. The following are the possible values:

- `ordering`: The initial state of a hosted connection provisioned on an interconnect. The connection stays in the ordering state until the owner of the hosted connection confirms or declines the connection order.
- `requested`: The initial state of a standard connection. The connection stays in the requested state until the Letter of Authorization (LOA) is sent to the customer.
- `pending`: The connection has been approved and is being initialized.
- `available`: The network link is up and the connection is ready for use.
- `down`: The network link is down.

- `deleting`: The connection is being deleted.
- `deleted`: The connection has been deleted.
- `rejected`: A hosted connection in the `ordering` state enters the `rejected` state if it is deleted by the customer.

Type: String

Valid Values: `ordering | requested | pending | available | down | deleting | deleted | rejected`

hasLogicalRedundancy (p. 22)

Indicates whether the connection supports a secondary BGP peer in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

jumboFrameCapable (p. 22)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

lagId (p. 22)

The ID of the LAG.

Type: String

loaIssueTime (p. 22)

The time of the most recent call to DescribeLoa (p. 113) for this connection.

Type: Timestamp

location (p. 22)

The location of the connection.

Type: String

ownerAccount (p. 22)

The ID of the AWS account that owns the connection.

Type: String

partnerName (p. 22)

The name of the AWS Direct Connect service provider associated with the connection.

Type: String

region (p. 22)

The AWS Region where the connection is located.

Type: String

vlan (p. 22)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

AssociateHostedConnection

Associates a hosted connection and its virtual interfaces with a link aggregation group (LAG) or interconnect. If the target interconnect or LAG has an existing hosted connection with a conflicting VLAN number or IP address, the operation fails. This action temporarily interrupts the hosted connection's connectivity to AWS as it is being migrated.

Note
Intended for use by AWS Direct Connect partners only.

Request Syntax

```
{
   "connectionId": "string",
   "parentConnectionId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 26)

The ID of the hosted connection.

Type: String

Required: Yes

parentConnectionId (p. 26)

The ID of the interconnect or the LAG.

Type: String

Required: Yes

Response Syntax

```
{
   "awsDevice": "string",
   "awsDeviceV2": "string",
   "bandwidth": "string",
   "connectionId": "string",
   "connectionName": "string",
   "connectionState": "string",
   "hasLogicalRedundancy": "string",
   "jumboFrameCapable": boolean,
   "lagId": "string",
   "loaIssueTime": number,
   "location": "string",
   "ownerAccount": "string",
   "partnerName": "string",
   "region": "string",
```

```
   "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

awsDevice (p. 26)

This parameter has been deprecated.

The Direct Connect endpoint on which the physical connection terminates.

Type: String

awsDeviceV2 (p. 26)

The Direct Connect endpoint on which the physical connection terminates.

Type: String

bandwidth (p. 26)

The bandwidth of the connection.

Type: String

connectionId (p. 26)

The ID of the connection.

Type: String

connectionName (p. 26)

The name of the connection.

Type: String

connectionState (p. 26)

The state of the connection. The following are the possible values:

- `ordering`: The initial state of a hosted connection provisioned on an interconnect. The connection stays in the ordering state until the owner of the hosted connection confirms or declines the connection order.
- `requested`: The initial state of a standard connection. The connection stays in the requested state until the Letter of Authorization (LOA) is sent to the customer.
- `pending`: The connection has been approved and is being initialized.
- `available`: The network link is up and the connection is ready for use.
- `down`: The network link is down.
- `deleting`: The connection is being deleted.
- `deleted`: The connection has been deleted.
- `rejected`: A hosted connection in the `ordering` state enters the `rejected` state if it is deleted by the customer.

Type: String

Valid Values: `ordering | requested | pending | available | down | deleting | deleted | rejected`

hasLogicalRedundancy (p. 26)

Indicates whether the connection supports a secondary BGP peer in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

jumboFrameCapable (p. 26)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

lagId (p. 26)

The ID of the LAG.

Type: String

loaIssueTime (p. 26)

The time of the most recent call to DescribeLoa (p. 113) for this connection.

Type: Timestamp

location (p. 26)

The location of the connection.

Type: String

ownerAccount (p. 26)

The ID of the AWS account that owns the connection.

Type: String

partnerName (p. 26)

The name of the AWS Direct Connect service provider associated with the connection.

Type: String

region (p. 26)

The AWS Region where the connection is located.

Type: String

vlan (p. 26)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

AssociateVirtualInterface

Associates a virtual interface with a specified link aggregation group (LAG) or connection. Connectivity to AWS is temporarily interrupted as the virtual interface is being migrated. If the target connection or LAG has an associated virtual interface with a conflicting VLAN number or a conflicting IP address, the operation fails.

Virtual interfaces associated with a hosted connection cannot be associated with a LAG; hosted connections must be migrated along with their virtual interfaces using AssociateHostedConnection (p. 26).

To reassociate a virtual interface to a new connection or LAG, the requester must own either the virtual interface itself or the connection to which the virtual interface is currently associated. Additionally, the requester must own the connection or LAG for the association.

Request Syntax

```
{
   "connectionId": "string",
   "virtualInterfaceId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 30)

The ID of the LAG or connection.

Type: String

Required: Yes

virtualInterfaceId (p. 30)

The ID of the virtual interface.

Type: String

Required: Yes

Response Syntax

```
{
   "addressFamily": "string",
   "amazonAddress": "string",
   "amazonSideAsn": number,
   "asn": number,
   "authKey": "string",
   "awsDeviceV2": "string",
   "bgpPeers": [
      {
```

```
            "addressFamily": "string",
            "amazonAddress": "string",
            "asn": number,
            "authKey": "string",
            "awsDeviceV2": "string",
            "bgpPeerId": "string",
            "bgpPeerState": "string",
            "bgpStatus": "string",
            "customerAddress": "string"
         }
      ],
      "connectionId": "string",
      "customerAddress": "string",
      "customerRouterConfig": "string",
      "directConnectGatewayId": "string",
      "jumboFrameCapable": boolean,
      "location": "string",
      "mtu": number,
      "ownerAccount": "string",
      "region": "string",
      "routeFilterPrefixes": [
         {
            "cidr": "string"
         }
      ],
      "virtualGatewayId": "string",
      "virtualInterfaceId": "string",
      "virtualInterfaceName": "string",
      "virtualInterfaceState": "string",
      "virtualInterfaceType": "string",
      "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

addressFamily (p. 30)

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

amazonAddress (p. 30)

The IP address assigned to the Amazon interface.

Type: String

amazonSideAsn (p. 30)

The autonomous system number (ASN) for the Amazon side of the connection.

Type: Long

asn (p. 30)

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

authKey (p. 30)

The authentication key for BGP configuration.

Type: String

awsDeviceV2 (p. 30)

The Direct Connect endpoint on which the virtual interface terminates.

Type: String

bgpPeers (p. 30)

The BGP peers configured on this virtual interface.

Type: Array of BGPPeer (p. 142) objects

connectionId (p. 30)

The ID of the connection.

Type: String

customerAddress (p. 30)

The IP address assigned to the customer interface.

Type: String

customerRouterConfig (p. 30)

The customer router configuration.

Type: String

directConnectGatewayId (p. 30)

The ID of the Direct Connect gateway.

Type: String

jumboFrameCapable (p. 30)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

location (p. 30)

The location of the connection.

Type: String

mtu (p. 30)

The maximum transmission unit (MTU), in bytes. The supported values are 1500 and 9001. The default value is 1500.

Type: Integer

ownerAccount (p. 30)

The ID of the AWS account that owns the virtual interface.

Type: String

region (p. 30)

The AWS Region where the virtual interface is located.

Type: String

routeFilterPrefixes (p. 30)

The routes to be advertised to the AWS network in this Region. Applies to public virtual interfaces.

Type: Array of RouteFilterPrefix (p. 170) objects

virtualGatewayId (p. 30)

The ID of the virtual private gateway. Applies only to private virtual interfaces.

Type: String

virtualInterfaceId (p. 30)

The ID of the virtual interface.

Type: String

virtualInterfaceName (p. 30)

The name of the virtual interface assigned by the customer network.

Type: String

virtualInterfaceState (p. 30)

The state of the virtual interface. The following are the possible values:

- `confirming`: The creation of the virtual interface is pending confirmation from the virtual interface owner. If the owner of the virtual interface is different from the owner of the connection on which it is provisioned, then the virtual interface will remain in this state until it is confirmed by the virtual interface owner.
- `verifying`: This state only applies to public virtual interfaces. Each public virtual interface needs validation before the virtual interface can be created.
- `pending`: A virtual interface is in this state from the time that it is created until the virtual interface is ready to forward traffic.
- `available`: A virtual interface that is able to forward traffic.
- `down`: A virtual interface that is BGP down.
- `deleting`: A virtual interface is in this state immediately after calling DeleteVirtualInterface (p. 89) until it can no longer forward traffic.
- `deleted`: A virtual interface that cannot forward traffic.
- `rejected`: The virtual interface owner has declined creation of the virtual interface. If a virtual interface in the `Confirming` state is deleted by the virtual interface owner, the virtual interface enters the `Rejected` state.

Type: String

Valid Values: `confirming | verifying | pending | available | down | deleting | deleted | rejected`

virtualInterfaceType (p. 30)

The type of virtual interface. The possible values are `private` and `public`.

Type: String

vlan (p. 30)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

ConfirmConnection

Confirms the creation of the specified hosted connection on an interconnect.

Upon creation, the hosted connection is initially in the `Ordering` state, and remains in this state until the owner confirms creation of the hosted connection.

Request Syntax

```
{
   "connectionId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 35)

The ID of the hosted connection.

Type: String

Required: Yes

Response Syntax

```
{
   "connectionState": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

connectionState (p. 35)

The state of the connection. The following are the possible values:

- `ordering`: The initial state of a hosted connection provisioned on an interconnect. The connection stays in the ordering state until the owner of the hosted connection confirms or declines the connection order.
- `requested`: The initial state of a standard connection. The connection stays in the requested state until the Letter of Authorization (LOA) is sent to the customer.
- `pending`: The connection has been approved and is being initialized.
- `available`: The network link is up and the connection is ready for use.
- `down`: The network link is down.
- `deleting`: The connection is being deleted.

- `deleted`: The connection has been deleted.
- `rejected`: A hosted connection in the `ordering` state enters the `rejected` state if it is deleted by the customer.

Type: String

Valid Values: `ordering | requested | pending | available | down | deleting | deleted | rejected`

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

ConfirmPrivateVirtualInterface

Accepts ownership of a private virtual interface created by another AWS account.

After the virtual interface owner makes this call, the virtual interface is created and attached to the specified virtual private gateway or Direct Connect gateway, and is made available to handle traffic.

Request Syntax

```
{
   "directConnectGatewayId": "string",
   "virtualGatewayId": "string",
   "virtualInterfaceId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 37)

The ID of the Direct Connect gateway.

Type: String

Required: No

virtualGatewayId (p. 37)

The ID of the virtual private gateway.

Type: String

Required: No

virtualInterfaceId (p. 37)

The ID of the virtual interface.

Type: String

Required: Yes

Response Syntax

```
{
   "virtualInterfaceState": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

virtualInterfaceState (p. 37)

The state of the virtual interface. The following are the possible values:

- `confirming`: The creation of the virtual interface is pending confirmation from the virtual interface owner. If the owner of the virtual interface is different from the owner of the connection on which it is provisioned, then the virtual interface will remain in this state until it is confirmed by the virtual interface owner.
- `verifying`: This state only applies to public virtual interfaces. Each public virtual interface needs validation before the virtual interface can be created.
- `pending`: A virtual interface is in this state from the time that it is created until the virtual interface is ready to forward traffic.
- `available`: A virtual interface that is able to forward traffic.
- `down`: A virtual interface that is BGP down.
- `deleting`: A virtual interface is in this state immediately after calling DeleteVirtualInterface (p. 89) until it can no longer forward traffic.
- `deleted`: A virtual interface that cannot forward traffic.
- `rejected`: The virtual interface owner has declined creation of the virtual interface. If a virtual interface in the `Confirming` state is deleted by the virtual interface owner, the virtual interface enters the `Rejected` state.

Type: String

Valid Values: `confirming | verifying | pending | available | down | deleting | deleted | rejected`

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript

- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

ConfirmPublicVirtualInterface

Accepts ownership of a public virtual interface created by another AWS account.

After the virtual interface owner makes this call, the specified virtual interface is created and made available to handle traffic.

Request Syntax

```
{
   "virtualInterfaceId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

virtualInterfaceId (p. 40)

The ID of the virtual interface.

Type: String

Required: Yes

Response Syntax

```
{
   "virtualInterfaceState": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

virtualInterfaceState (p. 40)

The state of the virtual interface. The following are the possible values:

- `confirming`: The creation of the virtual interface is pending confirmation from the virtual interface owner. If the owner of the virtual interface is different from the owner of the connection on which it is provisioned, then the virtual interface will remain in this state until it is confirmed by the virtual interface owner.
- `verifying`: This state only applies to public virtual interfaces. Each public virtual interface needs validation before the virtual interface can be created.
- `pending`: A virtual interface is in this state from the time that it is created until the virtual interface is ready to forward traffic.
- `available`: A virtual interface that is able to forward traffic.

- `down`: A virtual interface that is BGP down.
- `deleting`: A virtual interface is in this state immediately after calling DeleteVirtualInterface (p. 89) until it can no longer forward traffic.
- `deleted`: A virtual interface that cannot forward traffic.
- `rejected`: The virtual interface owner has declined creation of the virtual interface. If a virtual interface in the `Confirming` state is deleted by the virtual interface owner, the virtual interface enters the `Rejected` state.

Type: String

Valid Values: `confirming | verifying | pending | available | down | deleting | deleted | rejected`

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

CreateBGPPeer

Creates a BGP peer on the specified virtual interface.

You must create a BGP peer for the corresponding address family (IPv4/IPv6) in order to access AWS resources that also use that address family.

If logical redundancy is not supported by the connection, interconnect, or LAG, the BGP peer cannot be in the same address family as an existing BGP peer on the virtual interface.

When creating a IPv6 BGP peer, omit the Amazon address and customer address. IPv6 addresses are automatically assigned from the Amazon pool of IPv6 addresses; you cannot specify custom IPv6 addresses.

For a public virtual interface, the Autonomous System Number (ASN) must be private or already whitelisted for the virtual interface.

Request Syntax

```
{
   "newBGPPeer": {
      "addressFamily": "string",
      "amazonAddress": "string",
      "asn": number,
      "authKey": "string",
      "customerAddress": "string"
   },
   "virtualInterfaceId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

newBGPPeer (p. 42)

Information about the BGP peer.

Type: NewBGPPeer (p. 160) object

Required: No

virtualInterfaceId (p. 42)

The ID of the virtual interface.

Type: String

Required: No

Response Syntax

```
{
   "virtualInterface": {
```

```
      "addressFamily": "string",
      "amazonAddress": "string",
      "amazonSideAsn": number,
      "asn": number,
      "authKey": "string",
      "awsDeviceV2": "string",
      "bgpPeers": [
         {
            "addressFamily": "string",
            "amazonAddress": "string",
            "asn": number,
            "authKey": "string",
            "awsDeviceV2": "string",
            "bgpPeerId": "string",
            "bgpPeerState": "string",
            "bgpStatus": "string",
            "customerAddress": "string"
         }
      ],
      "connectionId": "string",
      "customerAddress": "string",
      "customerRouterConfig": "string",
      "directConnectGatewayId": "string",
      "jumboFrameCapable": boolean,
      "location": "string",
      "mtu": number,
      "ownerAccount": "string",
      "region": "string",
      "routeFilterPrefixes": [
         {
            "cidr": "string"
         }
      ],
      "virtualGatewayId": "string",
      "virtualInterfaceId": "string",
      "virtualInterfaceName": "string",
      "virtualInterfaceState": "string",
      "virtualInterfaceType": "string",
      "vlan": number
   }
}
```


Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

virtualInterface (p. 42)

The virtual interface.

Type: VirtualInterface (p. 173) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

CreateConnection

Creates a connection between a customer network and a specific AWS Direct Connect location.

A connection links your internal network to an AWS Direct Connect location over a standard Ethernet fiber-optic cable. One end of the cable is connected to your router, the other to an AWS Direct Connect router.

To find the locations for your Region, use DescribeLocations (p. 115).

You can automatically add the new connection to a link aggregation group (LAG) by specifying a LAG ID in the request. This ensures that the new connection is allocated on the same AWS Direct Connect endpoint that hosts the specified LAG. If there are no available ports on the endpoint, the request fails and no connection is created.

Request Syntax

```
{
   "bandwidth": "string",
   "connectionName": "string",
   "lagId": "string",
   "location": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

bandwidth (p. 45)

The bandwidth of the connection.

Type: String

Required: Yes

connectionName (p. 45)

The name of the connection.

Type: String

Required: Yes

lagId (p. 45)

The ID of the LAG.

Type: String

Required: No

location (p. 45)

The location of the connection.

Type: String

Required: Yes

Response Syntax

```
{
   "awsDevice": "string",
   "awsDeviceV2": "string",
   "bandwidth": "string",
   "connectionId": "string",
   "connectionName": "string",
   "connectionState": "string",
   "hasLogicalRedundancy": "string",
   "jumboFrameCapable": boolean,
   "lagId": "string",
   "loaIssueTime": number,
   "location": "string",
   "ownerAccount": "string",
   "partnerName": "string",
   "region": "string",
   "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

awsDevice (p. 46)

This parameter has been deprecated.

Type: String

awsDeviceV2 (p. 46)

The Direct Connect endpoint on which the physical connection terminates.

Type: String

bandwidth (p. 46)

The bandwidth of the connection.

Type: String

connectionId (p. 46)

The ID of the connection.

Type: String

connectionName (p. 46)

The name of the connection.

Type: String

connectionState (p. 46)

The state of the connection. The following are the possible values:

- `ordering`: The initial state of a hosted connection provisioned on an interconnect. The connection stays in the ordering state until the owner of the hosted connection confirms or declines the connection order.
- `requested`: The initial state of a standard connection. The connection stays in the requested state until the Letter of Authorization (LOA) is sent to the customer.
- `pending`: The connection has been approved and is being initialized.
- `available`: The network link is up and the connection is ready for use.
- `down`: The network link is down.
- `deleting`: The connection is being deleted.
- `deleted`: The connection has been deleted.
- `rejected`: A hosted connection in the `ordering` state enters the `rejected` state if it is deleted by the customer.

Type: String

Valid Values: `ordering | requested | pending | available | down | deleting | deleted | rejected`

hasLogicalRedundancy (p. 46)

Indicates whether the connection supports a secondary BGP peer in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

jumboFrameCapable (p. 46)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

lagId (p. 46)

The ID of the LAG.

Type: String

loaIssueTime (p. 46)

The time of the most recent call to DescribeLoa (p. 113) for this connection.

Type: Timestamp

location (p. 46)

The location of the connection.

Type: String

ownerAccount (p. 46)

The ID of the AWS account that owns the connection.

Type: String

partnerName (p. 46)

The name of the AWS Direct Connect service provider associated with the connection.

Type: String

region (p. 46)

The AWS Region where the connection is located.

Type: String

vlan (p. 46)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

CreateDirectConnectGateway

Creates a Direct Connect gateway, which is an intermediate object that enables you to connect a set of virtual interfaces and virtual private gateways. A Direct Connect gateway is global and visible in any AWS Region after it is created. The virtual interfaces and virtual private gateways that are connected through a Direct Connect gateway can be in different AWS Regions. This enables you to connect to a VPC in any Region, regardless of the Region in which the virtual interfaces are located, and pass traffic between them.

Request Syntax

```
{
   "amazonSideAsn": number,
   "directConnectGatewayName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

amazonSideAsn (p. 49)

The autonomous system number (ASN) for Border Gateway Protocol (BGP) to be configured on the Amazon side of the connection. The ASN must be in the private range of 64,512 to 65,534 or 4,200,000,000 to 4,294,967,294. The default is 64512.

Type: Long

Required: No

directConnectGatewayName (p. 49)

The name of the Direct Connect gateway.

Type: String

Required: Yes

Response Syntax

```
{
   "directConnectGateway": {
      "amazonSideAsn": number,
      "directConnectGatewayId": "string",
      "directConnectGatewayName": "string",
      "directConnectGatewayState": "string",
      "ownerAccount": "string",
      "stateChangeError": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

directConnectGateway (p. 49)

The Direct Connect gateway.

Type: DirectConnectGateway (p. 147) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

CreateDirectConnectGatewayAssociation

Creates an association between a Direct Connect gateway and a virtual private gateway. The virtual private gateway must be attached to a VPC and must not be associated with another Direct Connect gateway.

Request Syntax

```
{
   "directConnectGatewayId": "string",
   "virtualGatewayId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 51)

The ID of the Direct Connect gateway.

Type: String

Required: Yes

virtualGatewayId (p. 51)

The ID of the virtual private gateway.

Type: String

Required: Yes

Response Syntax

```
{
   "directConnectGatewayAssociation": {
      "associationState": "string",
      "directConnectGatewayId": "string",
      "stateChangeError": "string",
      "virtualGatewayId": "string",
      "virtualGatewayOwnerAccount": "string",
      "virtualGatewayRegion": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

directConnectGatewayAssociation (p. 51)

The association to be created.

Type: DirectConnectGatewayAssociation (p. 149) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

CreateInterconnect

Creates an interconnect between an AWS Direct Connect partner's network and a specific AWS Direct Connect location.

An interconnect is a connection which is capable of hosting other connections. The partner can use an interconnect to provide sub-1Gbps AWS Direct Connect service to tier 2 customers who do not have their own connections. Like a standard connection, an interconnect links the partner's network to an AWS Direct Connect location over a standard Ethernet fiber-optic cable. One end is connected to the partner's router, the other to an AWS Direct Connect router.

You can automatically add the new interconnect to a link aggregation group (LAG) by specifying a LAG ID in the request. This ensures that the new interconnect is allocated on the same AWS Direct Connect endpoint that hosts the specified LAG. If there are no available ports on the endpoint, the request fails and no interconnect is created.

For each end customer, the AWS Direct Connect partner provisions a connection on their interconnect by calling AllocateConnectionOnInterconnect (p. 4). The end customer can then connect to AWS resources by creating a virtual interface on their connection, using the VLAN assigned to them by the partner.

Note
Intended for use by AWS Direct Connect partners only.

Request Syntax

```
{
   "bandwidth": "string",
   "interconnectName": "string",
   "lagId": "string",
   "location": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

bandwidth (p. 53)

The port bandwidth, in Gbps. The possible values are 1 and 10.

Type: String

Required: Yes

interconnectName (p. 53)

The name of the interconnect.

Type: String

Required: Yes

lagId (p. 53)

The ID of the LAG.

Type: String

Required: No

location (p. 53)

The location of the interconnect.

Type: String

Required: Yes

Response Syntax

```
{
   "awsDevice": "string",
   "awsDeviceV2": "string",
   "bandwidth": "string",
   "hasLogicalRedundancy": "string",
   "interconnectId": "string",
   "interconnectName": "string",
   "interconnectState": "string",
   "jumboFrameCapable": boolean,
   "lagId": "string",
   "loaIssueTime": number,
   "location": "string",
   "region": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

awsDevice (p. 54)

This parameter has been deprecated.

The Direct Connect endpoint on which the physical connection terminates.

Type: String

awsDeviceV2 (p. 54)

The Direct Connect endpoint on which the physical connection terminates.

Type: String

bandwidth (p. 54)

The bandwidth of the connection.

Type: String

hasLogicalRedundancy (p. 54)

Indicates whether the interconnect supports a secondary BGP in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

interconnectId (p. 54)

The ID of the interconnect.

Type: String

interconnectName (p. 54)

The name of the interconnect.

Type: String

interconnectState (p. 54)

The state of the interconnect. The following are the possible values:

- `requested`: The initial state of an interconnect. The interconnect stays in the requested state until the Letter of Authorization (LOA) is sent to the customer.
- `pending`: The interconnect is approved, and is being initialized.
- `available`: The network link is up, and the interconnect is ready for use.
- `down`: The network link is down.
- `deleting`: The interconnect is being deleted.
- `deleted`: The interconnect is deleted.

Type: String

Valid Values: `requested | pending | available | down | deleting | deleted`

jumboFrameCapable (p. 54)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

lagId (p. 54)

The ID of the LAG.

Type: String

loaIssueTime (p. 54)

The time of the most recent call to DescribeLoa (p. 113) for this connection.

Type: Timestamp

location (p. 54)

The location of the connection.

Type: String

region (p. 54)

The AWS Region where the connection is located.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

CreateLag

Creates a link aggregation group (LAG) with the specified number of bundled physical connections between the customer network and a specific AWS Direct Connect location. A LAG is a logical interface that uses the Link Aggregation Control Protocol (LACP) to aggregate multiple interfaces, enabling you to treat them as a single interface.

All connections in a LAG must use the same bandwidth and must terminate at the same AWS Direct Connect endpoint.

You can have up to 10 connections per LAG. Regardless of this limit, if you request more connections for the LAG than AWS Direct Connect can allocate on a single endpoint, no LAG is created.

You can specify an existing physical connection or interconnect to include in the LAG (which counts towards the total number of connections). Doing so interrupts the current physical connection or hosted connections, and re-establishes them as a member of the LAG. The LAG will be created on the same AWS Direct Connect endpoint to which the connection terminates. Any virtual interfaces associated with the connection are automatically disassociated and re-associated with the LAG. The connection ID does not change.

If the AWS account used to create a LAG is a registered AWS Direct Connect partner, the LAG is automatically enabled to host sub-connections. For a LAG owned by a partner, any associated virtual interfaces cannot be directly configured.

Request Syntax

```
{
   "connectionId": "string",
   "connectionsBandwidth": "string",
   "lagName": "string",
   "location": "string",
   "numberOfConnections": number
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 57)

The ID of an existing connection to migrate to the LAG.

Type: String

Required: No

connectionsBandwidth (p. 57)

The bandwidth of the individual physical connections bundled by the LAG. The possible values are 1Gbps and 10Gbps.

Type: String

Required: Yes

lagName (p. 57)

The name of the LAG.

Type: String

Required: Yes

location (p. 57)

The location for the LAG.

Type: String

Required: Yes

numberOfConnections (p. 57)

The number of physical connections initially provisioned and bundled by the LAG.

Type: Integer

Required: Yes

Response Syntax

```
{
   "allowsHostedConnections": boolean,
   "awsDevice": "string",
   "awsDeviceV2": "string",
   "connections": [
      {
         "awsDevice": "string",
         "awsDeviceV2": "string",
         "bandwidth": "string",
         "connectionId": "string",
         "connectionName": "string",
         "connectionState": "string",
         "hasLogicalRedundancy": "string",
         "jumboFrameCapable": boolean,
         "lagId": "string",
         "loaIssueTime": number,
         "location": "string",
         "ownerAccount": "string",
         "partnerName": "string",
         "region": "string",
         "vlan": number
      }
   ],
   "connectionsBandwidth": "string",
   "hasLogicalRedundancy": "string",
   "jumboFrameCapable": boolean,
   "lagId": "string",
   "lagName": "string",
   "lagState": "string",
   "location": "string",
   "minimumLinks": number,
   "numberOfConnections": number,
   "ownerAccount": "string",
   "region": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

allowsHostedConnections (p. 58)

Indicates whether the LAG can host other connections.

Type: Boolean

awsDevice (p. 58)

This parameter has been deprecated.

The Direct Connect endpoint that hosts the LAG.

Type: String

awsDeviceV2 (p. 58)

The Direct Connect endpoint that hosts the LAG.

Type: String

connections (p. 58)

The connections bundled by the LAG.

Type: Array of Connection (p. 144) objects

connectionsBandwidth (p. 58)

The individual bandwidth of the physical connections bundled by the LAG. The possible values are 1Gbps and 10Gbps.

Type: String

hasLogicalRedundancy (p. 58)

Indicates whether the LAG supports a secondary BGP peer in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

jumboFrameCapable (p. 58)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

lagId (p. 58)

The ID of the LAG.

Type: String

lagName (p. 58)

The name of the LAG.

Type: String

lagState (p. 58)

The state of the LAG. The following are the possible values:

- `requested`: The initial state of a LAG. The LAG stays in the requested state until the Letter of Authorization (LOA) is available.
- `pending`: The LAG has been approved and is being initialized.
- `available`: The network link is established and the LAG is ready for use.
- `down`: The network link is down.
- `deleting`: The LAG is being deleted.
- `deleted`: The LAG is deleted.

Type: String

Valid Values: `requested | pending | available | down | deleting | deleted`

location (p. 58)

The location of the LAG.

Type: String

minimumLinks (p. 58)

The minimum number of physical connections that must be operational for the LAG itself to be operational.

Type: Integer

numberOfConnections (p. 58)

The number of physical connections bundled by the LAG, up to a maximum of 10.

Type: Integer

ownerAccount (p. 58)

The ID of the AWS account that owns the LAG.

Type: String

region (p. 58)

The AWS Region where the connection is located.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

CreatePrivateVirtualInterface

Creates a private virtual interface. A virtual interface is the VLAN that transports AWS Direct Connect traffic. A private virtual interface can be connected to either a Direct Connect gateway or a Virtual Private Gateway (VGW). Connecting the private virtual interface to a Direct Connect gateway enables the possibility for connecting to multiple VPCs, including VPCs in different AWS Regions. Connecting the private virtual interface to a VGW only provides access to a single VPC within the same Region.

Request Syntax

```
{
   "connectionId": "string",
   "newPrivateVirtualInterface": {
      "addressFamily": "string",
      "amazonAddress": "string",
      "asn": number,
      "authKey": "string",
      "customerAddress": "string",
      "directConnectGatewayId": "string",
      "mtu": number,
      "virtualGatewayId": "string",
      "virtualInterfaceName": "string",
      "vlan": number
   }
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 62)

The ID of the connection.

Type: String

Required: Yes

newPrivateVirtualInterface (p. 62)

Information about the private virtual interface.

Type: NewPrivateVirtualInterface (p. 161) object

Required: Yes

Response Syntax

```
{
   "addressFamily": "string",
   "amazonAddress": "string",
   "amazonSideAsn": number,
   "asn": number,
   "authKey": "string",
```

```
   "awsDeviceV2": "string",
   "bgpPeers": [
      {
         "addressFamily": "string",
         "amazonAddress": "string",
         "asn": number,
         "authKey": "string",
         "awsDeviceV2": "string",
         "bgpPeerId": "string",
         "bgpPeerState": "string",
         "bgpStatus": "string",
         "customerAddress": "string"
      }
   ],
   "connectionId": "string",
   "customerAddress": "string",
   "customerRouterConfig": "string",
   "directConnectGatewayId": "string",
   "jumboFrameCapable": boolean,
   "location": "string",
   "mtu": number,
   "ownerAccount": "string",
   "region": "string",
   "routeFilterPrefixes": [
      {
         "cidr": "string"
      }
   ],
   "virtualGatewayId": "string",
   "virtualInterfaceId": "string",
   "virtualInterfaceName": "string",
   "virtualInterfaceState": "string",
   "virtualInterfaceType": "string",
   "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

addressFamily (p. 62)

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

amazonAddress (p. 62)

The IP address assigned to the Amazon interface.

Type: String

amazonSideAsn (p. 62)

The autonomous system number (ASN) for the Amazon side of the connection.

Type: Long

asn (p. 62)

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

authKey (p. 62)

The authentication key for BGP configuration.

Type: String

awsDeviceV2 (p. 62)

The Direct Connect endpoint on which the virtual interface terminates.

Type: String

bgpPeers (p. 62)

The BGP peers configured on this virtual interface.

Type: Array of BGPPeer (p. 142) objects

connectionId (p. 62)

The ID of the connection.

Type: String

customerAddress (p. 62)

The IP address assigned to the customer interface.

Type: String

customerRouterConfig (p. 62)

The customer router configuration.

Type: String

directConnectGatewayId (p. 62)

The ID of the Direct Connect gateway.

Type: String

jumboFrameCapable (p. 62)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

location (p. 62)

The location of the connection.

Type: String

mtu (p. 62)

The maximum transmission unit (MTU), in bytes. The supported values are 1500 and 9001. The default value is 1500.

Type: Integer

ownerAccount (p. 62)

The ID of the AWS account that owns the virtual interface.

Type: String

region (p. 62)

The AWS Region where the virtual interface is located.

Type: String

routeFilterPrefixes (p. 62)

The routes to be advertised to the AWS network in this Region. Applies to public virtual interfaces.

Type: Array of RouteFilterPrefix (p. 170) objects

virtualGatewayId (p. 62)

The ID of the virtual private gateway. Applies only to private virtual interfaces.

Type: String

virtualInterfaceId (p. 62)

The ID of the virtual interface.

Type: String

virtualInterfaceName (p. 62)

The name of the virtual interface assigned by the customer network.

Type: String

virtualInterfaceState (p. 62)

The state of the virtual interface. The following are the possible values:

- `confirming`: The creation of the virtual interface is pending confirmation from the virtual interface owner. If the owner of the virtual interface is different from the owner of the connection on which it is provisioned, then the virtual interface will remain in this state until it is confirmed by the virtual interface owner.
- `verifying`: This state only applies to public virtual interfaces. Each public virtual interface needs validation before the virtual interface can be created.
- `pending`: A virtual interface is in this state from the time that it is created until the virtual interface is ready to forward traffic.
- `available`: A virtual interface that is able to forward traffic.
- `down`: A virtual interface that is BGP down.
- `deleting`: A virtual interface is in this state immediately after calling DeleteVirtualInterface (p. 89) until it can no longer forward traffic.
- `deleted`: A virtual interface that cannot forward traffic.
- `rejected`: The virtual interface owner has declined creation of the virtual interface. If a virtual interface in the `Confirming` state is deleted by the virtual interface owner, the virtual interface enters the `Rejected` state.

Type: String

Valid Values: `confirming | verifying | pending | available | down | deleting | deleted | rejected`

virtualInterfaceType (p. 62)

The type of virtual interface. The possible values are `private` and `public`.

Type: String

vlan (p. 62)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

CreatePublicVirtualInterface

Creates a public virtual interface. A virtual interface is the VLAN that transports AWS Direct Connect traffic. A public virtual interface supports sending traffic to public services of AWS such as Amazon S3.

When creating an IPv6 public virtual interface (`addressFamily` is `ipv6`), leave the `customer` and `amazon` address fields blank to use auto-assigned IPv6 space. Custom IPv6 addresses are not supported.

Request Syntax

```
{
   "connectionId": "string",
   "newPublicVirtualInterface": {
      "addressFamily": "string",
      "amazonAddress": "string",
      "asn": number,
      "authKey": "string",
      "customerAddress": "string",
      "routeFilterPrefixes": [
         {
            "cidr": "string"
         }
      ],
      "virtualInterfaceName": "string",
      "vlan": number
   }
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 67)

The ID of the connection.

Type: String

Required: Yes

newPublicVirtualInterface (p. 67)

Information about the public virtual interface.

Type: NewPublicVirtualInterface (p. 165) object

Required: Yes

Response Syntax

```
{
   "addressFamily": "string",
   "amazonAddress": "string",
```

```
   "amazonSideAsn": number,
   "asn": number,
   "authKey": "string",
   "awsDeviceV2": "string",
   "bgpPeers": [
      {
         "addressFamily": "string",
         "amazonAddress": "string",
         "asn": number,
         "authKey": "string",
         "awsDeviceV2": "string",
         "bgpPeerId": "string",
         "bgpPeerState": "string",
         "bgpStatus": "string",
         "customerAddress": "string"
      }
   ],
   "connectionId": "string",
   "customerAddress": "string",
   "customerRouterConfig": "string",
   "directConnectGatewayId": "string",
   "jumboFrameCapable": boolean,
   "location": "string",
   "mtu": number,
   "ownerAccount": "string",
   "region": "string",
   "routeFilterPrefixes": [
      {
         "cidr": "string"
      }
   ],
   "virtualGatewayId": "string",
   "virtualInterfaceId": "string",
   "virtualInterfaceName": "string",
   "virtualInterfaceState": "string",
   "virtualInterfaceType": "string",
   "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

addressFamily (p. 67)

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

amazonAddress (p. 67)

The IP address assigned to the Amazon interface.

Type: String

amazonSideAsn (p. 67)

The autonomous system number (ASN) for the Amazon side of the connection.

Type: Long

asn (p. 67)

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

authKey (p. 67)

The authentication key for BGP configuration.

Type: String

awsDeviceV2 (p. 67)

The Direct Connect endpoint on which the virtual interface terminates.

Type: String

bgpPeers (p. 67)

The BGP peers configured on this virtual interface.

Type: Array of BGPPeer (p. 142) objects

connectionId (p. 67)

The ID of the connection.

Type: String

customerAddress (p. 67)

The IP address assigned to the customer interface.

Type: String

customerRouterConfig (p. 67)

The customer router configuration.

Type: String

directConnectGatewayId (p. 67)

The ID of the Direct Connect gateway.

Type: String

jumboFrameCapable (p. 67)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

location (p. 67)

The location of the connection.

Type: String

mtu (p. 67)

The maximum transmission unit (MTU), in bytes. The supported values are 1500 and 9001. The default value is 1500.

Type: Integer

ownerAccount (p. 67)

The ID of the AWS account that owns the virtual interface.

Type: String

region (p. 67)

The AWS Region where the virtual interface is located.

Type: String

routeFilterPrefixes (p. 67)

The routes to be advertised to the AWS network in this Region. Applies to public virtual interfaces.

Type: Array of RouteFilterPrefix (p. 170) objects

virtualGatewayId (p. 67)

The ID of the virtual private gateway. Applies only to private virtual interfaces.

Type: String

virtualInterfaceId (p. 67)

The ID of the virtual interface.

Type: String

virtualInterfaceName (p. 67)

The name of the virtual interface assigned by the customer network.

Type: String

virtualInterfaceState (p. 67)

The state of the virtual interface. The following are the possible values:

- `confirming`: The creation of the virtual interface is pending confirmation from the virtual interface owner. If the owner of the virtual interface is different from the owner of the connection on which it is provisioned, then the virtual interface will remain in this state until it is confirmed by the virtual interface owner.
- `verifying`: This state only applies to public virtual interfaces. Each public virtual interface needs validation before the virtual interface can be created.
- `pending`: A virtual interface is in this state from the time that it is created until the virtual interface is ready to forward traffic.
- `available`: A virtual interface that is able to forward traffic.
- `down`: A virtual interface that is BGP down.
- `deleting`: A virtual interface is in this state immediately after calling DeleteVirtualInterface (p. 89) until it can no longer forward traffic.
- `deleted`: A virtual interface that cannot forward traffic.
- `rejected`: The virtual interface owner has declined creation of the virtual interface. If a virtual interface in the `Confirming` state is deleted by the virtual interface owner, the virtual interface enters the `Rejected` state.

Type: String

Valid Values: `confirming | verifying | pending | available | down | deleting | deleted | rejected`

virtualInterfaceType (p. 67)

The type of virtual interface. The possible values are `private` and `public`.

Type: String

vlan (p. 67)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DeleteBGPPeer

Deletes the specified BGP peer on the specified virtual interface with the specified customer address and ASN.

You cannot delete the last BGP peer from a virtual interface.

Request Syntax

```
{
   "asn": number,
   "bgpPeerId": "string",
   "customerAddress": "string",
   "virtualInterfaceId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

asn (p. 72)

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

Required: No

bgpPeerId (p. 72)

The ID of the BGP peer.

Type: String

Required: No

customerAddress (p. 72)

The IP address assigned to the customer interface.

Type: String

Required: No

virtualInterfaceId (p. 72)

The ID of the virtual interface.

Type: String

Required: No

Response Syntax

```
{
   "virtualInterface": {
```

```
      "addressFamily": "string",
      "amazonAddress": "string",
      "amazonSideAsn": number,
      "asn": number,
      "authKey": "string",
      "awsDeviceV2": "string",
      "bgpPeers": [
         {
            "addressFamily": "string",
            "amazonAddress": "string",
            "asn": number,
            "authKey": "string",
            "awsDeviceV2": "string",
            "bgpPeerId": "string",
            "bgpPeerState": "string",
            "bgpStatus": "string",
            "customerAddress": "string"
         }
      ],
      "connectionId": "string",
      "customerAddress": "string",
      "customerRouterConfig": "string",
      "directConnectGatewayId": "string",
      "jumboFrameCapable": boolean,
      "location": "string",
      "mtu": number,
      "ownerAccount": "string",
      "region": "string",
      "routeFilterPrefixes": [
         {
            "cidr": "string"
         }
      ],
      "virtualGatewayId": "string",
      "virtualInterfaceId": "string",
      "virtualInterfaceName": "string",
      "virtualInterfaceState": "string",
      "virtualInterfaceType": "string",
      "vlan": number
   }
}
```


Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

virtualInterface (p. 72)

The virtual interface.

Type: VirtualInterface (p. 173) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DeleteConnection

Deletes the specified connection.

Deleting a connection only stops the AWS Direct Connect port hour and data transfer charges. If you are partnering with any third parties to connect with the AWS Direct Connect location, you must cancel your service with them separately.

Request Syntax

```
{
   "connectionId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 75)

The ID of the connection.

Type: String

Required: Yes

Response Syntax

```
{
   "awsDevice": "string",
   "awsDeviceV2": "string",
   "bandwidth": "string",
   "connectionId": "string",
   "connectionName": "string",
   "connectionState": "string",
   "hasLogicalRedundancy": "string",
   "jumboFrameCapable": boolean,
   "lagId": "string",
   "loaIssueTime": number,
   "location": "string",
   "ownerAccount": "string",
   "partnerName": "string",
   "region": "string",
   "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

awsDevice (p. 75)

This parameter has been deprecated.

The Direct Connect endpoint on which the physical connection terminates.

Type: String

awsDeviceV2 (p. 75)

The Direct Connect endpoint on which the physical connection terminates.

Type: String

bandwidth (p. 75)

The bandwidth of the connection.

Type: String

connectionId (p. 75)

The ID of the connection.

Type: String

connectionName (p. 75)

The name of the connection.

Type: String

connectionState (p. 75)

The state of the connection. The following are the possible values:

- `ordering`: The initial state of a hosted connection provisioned on an interconnect. The connection stays in the ordering state until the owner of the hosted connection confirms or declines the connection order.
- `requested`: The initial state of a standard connection. The connection stays in the requested state until the Letter of Authorization (LOA) is sent to the customer.
- `pending`: The connection has been approved and is being initialized.
- `available`: The network link is up and the connection is ready for use.
- `down`: The network link is down.
- `deleting`: The connection is being deleted.
- `deleted`: The connection has been deleted.
- `rejected`: A hosted connection in the `ordering` state enters the `rejected` state if it is deleted by the customer.

Type: String

Valid Values: `ordering | requested | pending | available | down | deleting | deleted | rejected`

hasLogicalRedundancy (p. 75)

Indicates whether the connection supports a secondary BGP peer in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

jumboFrameCapable (p. 75)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

lagId (p. 75)

The ID of the LAG.

Type: String

loaIssueTime (p. 75)

The time of the most recent call to DescribeLoa (p. 113) for this connection.

Type: Timestamp

location (p. 75)

The location of the connection.

Type: String

ownerAccount (p. 75)

The ID of the AWS account that owns the connection.

Type: String

partnerName (p. 75)

The name of the AWS Direct Connect service provider associated with the connection.

Type: String

region (p. 75)

The AWS Region where the connection is located.

Type: String

vlan (p. 75)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DeleteDirectConnectGateway

Deletes the specified Direct Connect gateway. You must first delete all virtual interfaces that are attached to the Direct Connect gateway and disassociate all virtual private gateways that are associated with the Direct Connect gateway.

Request Syntax

```
{
   "directConnectGatewayId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 79)

The ID of the Direct Connect gateway.

Type: String

Required: Yes

Response Syntax

```
{
   "directConnectGateway": {
      "amazonSideAsn": number,
      "directConnectGatewayId": "string",
      "directConnectGatewayName": "string",
      "directConnectGatewayState": "string",
      "ownerAccount": "string",
      "stateChangeError": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

directConnectGateway (p. 79)

The Direct Connect gateway.

Type: DirectConnectGateway (p. 147) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DeleteDirectConnectGatewayAssociation

Deletes the association between the specified Direct Connect gateway and virtual private gateway.

Request Syntax

```
{
   "directConnectGatewayId": "string",
   "virtualGatewayId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 81)

The ID of the Direct Connect gateway.

Type: String

Required: Yes

virtualGatewayId (p. 81)

The ID of the virtual private gateway.

Type: String

Required: Yes

Response Syntax

```
{
   "directConnectGatewayAssociation": {
      "associationState": "string",
      "directConnectGatewayId": "string",
      "stateChangeError": "string",
      "virtualGatewayId": "string",
      "virtualGatewayOwnerAccount": "string",
      "virtualGatewayRegion": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

directConnectGatewayAssociation (p. 81)

The association to be deleted.

Type: DirectConnectGatewayAssociation (p. 149) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DeleteInterconnect

Deletes the specified interconnect.

Note
Intended for use by AWS Direct Connect partners only.

Request Syntax

```
{
   "interconnectId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

interconnectId (p. 83)

The ID of the interconnect.

Type: String

Required: Yes

Response Syntax

```
{
   "interconnectState": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

interconnectState (p. 83)

The state of the interconnect. The following are the possible values:

- `requested`: The initial state of an interconnect. The interconnect stays in the requested state until the Letter of Authorization (LOA) is sent to the customer.
- `pending`: The interconnect is approved, and is being initialized.
- `available`: The network link is up, and the interconnect is ready for use.
- `down`: The network link is down.
- `deleting`: The interconnect is being deleted.
- `deleted`: The interconnect is deleted.

Type: String

Valid Values: `requested | pending | available | down | deleting | deleted`

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DeleteLag

Deletes the specified link aggregation group (LAG). You cannot delete a LAG if it has active virtual interfaces or hosted connections.

Request Syntax

```
{
   "lagId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

lagId (p. 85)

The ID of the LAG.

Type: String

Required: Yes

Response Syntax

```
{
   "allowsHostedConnections": boolean,
   "awsDevice": "string",
   "awsDeviceV2": "string",
   "connections": [
      {
         "awsDevice": "string",
         "awsDeviceV2": "string",
         "bandwidth": "string",
         "connectionId": "string",
         "connectionName": "string",
         "connectionState": "string",
         "hasLogicalRedundancy": "string",
         "jumboFrameCapable": boolean,
         "lagId": "string",
         "loaIssueTime": number,
         "location": "string",
         "ownerAccount": "string",
         "partnerName": "string",
         "region": "string",
         "vlan": number
      }
   ],
   "connectionsBandwidth": "string",
   "hasLogicalRedundancy": "string",
   "jumboFrameCapable": boolean,
   "lagId": "string",
   "lagName": "string",
   "lagState": "string",
```

```
   "location": "string",
   "minimumLinks": number,
   "numberOfConnections": number,
   "ownerAccount": "string",
   "region": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

allowsHostedConnections (p. 85)

Indicates whether the LAG can host other connections.

Type: Boolean

awsDevice (p. 85)

This parameter has been deprecated.

The Direct Connect endpoint that hosts the LAG.

Type: String

awsDeviceV2 (p. 85)

The Direct Connect endpoint that hosts the LAG.

Type: String

connections (p. 85)

The connections bundled by the LAG.

Type: Array of Connection (p. 144) objects

connectionsBandwidth (p. 85)

The individual bandwidth of the physical connections bundled by the LAG. The possible values are 1Gbps and 10Gbps.

Type: String

hasLogicalRedundancy (p. 85)

Indicates whether the LAG supports a secondary BGP peer in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

jumboFrameCapable (p. 85)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

lagId (p. 85)

The ID of the LAG.

Type: String

lagName (p. 85)

The name of the LAG.

Type: String

lagState (p. 85)

The state of the LAG. The following are the possible values:

- `requested`: The initial state of a LAG. The LAG stays in the requested state until the Letter of Authorization (LOA) is available.
- `pending`: The LAG has been approved and is being initialized.
- `available`: The network link is established and the LAG is ready for use.
- `down`: The network link is down.
- `deleting`: The LAG is being deleted.
- `deleted`: The LAG is deleted.

Type: String

Valid Values: `requested | pending | available | down | deleting | deleted`

location (p. 85)

The location of the LAG.

Type: String

minimumLinks (p. 85)

The minimum number of physical connections that must be operational for the LAG itself to be operational.

Type: Integer

numberOfConnections (p. 85)

The number of physical connections bundled by the LAG, up to a maximum of 10.

Type: Integer

ownerAccount (p. 85)

The ID of the AWS account that owns the LAG.

Type: String

region (p. 85)

The AWS Region where the connection is located.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DeleteVirtualInterface

Deletes a virtual interface.

Request Syntax

```
{
   "virtualInterfaceId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

virtualInterfaceId (p. 89)

The ID of the virtual interface.

Type: String

Required: Yes

Response Syntax

```
{
   "virtualInterfaceState": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

virtualInterfaceState (p. 89)

The state of the virtual interface. The following are the possible values:

- `confirming`: The creation of the virtual interface is pending confirmation from the virtual interface owner. If the owner of the virtual interface is different from the owner of the connection on which it is provisioned, then the virtual interface will remain in this state until it is confirmed by the virtual interface owner.
- `verifying`: This state only applies to public virtual interfaces. Each public virtual interface needs validation before the virtual interface can be created.
- `pending`: A virtual interface is in this state from the time that it is created until the virtual interface is ready to forward traffic.
- `available`: A virtual interface that is able to forward traffic.
- `down`: A virtual interface that is BGP down.
- `deleting`: A virtual interface is in this state immediately after calling DeleteVirtualInterface (p. 89) until it can no longer forward traffic.

- `deleted`: A virtual interface that cannot forward traffic.
- `rejected`: The virtual interface owner has declined creation of the virtual interface. If a virtual interface in the `Confirming` state is deleted by the virtual interface owner, the virtual interface enters the `Rejected` state.

Type: String

Valid Values: `confirming | verifying | pending | available | down | deleting | deleted | rejected`

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeConnectionLoa

Deprecated. Use DescribeLoa (p. 113) instead.

Gets the LOA-CFA for a connection.

The Letter of Authorization - Connecting Facility Assignment (LOA-CFA) is a document that your APN partner or service provider uses when establishing your cross connect to AWS at the colocation facility. For more information, see Requesting Cross Connects at AWS Direct Connect Locations in the *AWS Direct Connect User Guide*.

Request Syntax

```
{
   "connectionId": "string",
   "loaContentType": "string",
   "providerName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 91)

The ID of the connection.

Type: String

Required: Yes

loaContentType (p. 91)

The standard media type for the LOA-CFA document. The only supported value is application/pdf.

Type: String

Valid Values: `application/pdf`

Required: No

providerName (p. 91)

The name of the APN partner or service provider who establishes connectivity on your behalf. If you specify this parameter, the LOA-CFA lists the provider name alongside your company name as the requester of the cross connect.

Type: String

Required: No

Response Syntax

```
{
```

```
   "loa": {
      "loaContent": blob,
      "loaContentType": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

loa (p. 91)

The Letter of Authorization - Connecting Facility Assignment (LOA-CFA).

Type: Loa (p. 158) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeConnections

Displays the specified connection or all connections in this Region.

Request Syntax

```
{
   "connectionId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 93)

The ID of the connection.

Type: String

Required: No

Response Syntax

```
{
   "connections": [
      {
         "awsDevice": "string",
         "awsDeviceV2": "string",
         "bandwidth": "string",
         "connectionId": "string",
         "connectionName": "string",
         "connectionState": "string",
         "hasLogicalRedundancy": "string",
         "jumboFrameCapable": boolean,
         "lagId": "string",
         "loaIssueTime": number,
         "location": "string",
         "ownerAccount": "string",
         "partnerName": "string",
         "region": "string",
         "vlan": number
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

connections (p. 93)

The connections.

Type: Array of Connection (p. 144) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeConnectionsOnInterconnect

Deprecated. Use DescribeHostedConnections (p. 105) instead.

Lists the connections that have been provisioned on the specified interconnect.

Note
Intended for use by AWS Direct Connect partners only.

Request Syntax

```
{
   "interconnectId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

interconnectId (p. 95)

The ID of the interconnect.

Type: String

Required: Yes

Response Syntax

```
{
   "connections": [
      {
         "awsDevice": "string",
         "awsDeviceV2": "string",
         "bandwidth": "string",
         "connectionId": "string",
         "connectionName": "string",
         "connectionState": "string",
         "hasLogicalRedundancy": "string",
         "jumboFrameCapable": boolean,
         "lagId": "string",
         "loaIssueTime": number,
         "location": "string",
         "ownerAccount": "string",
         "partnerName": "string",
         "region": "string",
         "vlan": number
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

connections (p. 95)

The connections.

Type: Array of Connection (p. 144) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeDirectConnectGatewayAssociations

Lists the associations between your Direct Connect gateways and virtual private gateways. You must specify a Direct Connect gateway, a virtual private gateway, or both. If you specify a Direct Connect gateway, the response contains all virtual private gateways associated with the Direct Connect gateway. If you specify a virtual private gateway, the response contains all Direct Connect gateways associated with the virtual private gateway. If you specify both, the response contains the association between the Direct Connect gateway and the virtual private gateway.

Request Syntax

```
{
   "directConnectGatewayId": "string",
   "maxResults": number,
   "nextToken": "string",
   "virtualGatewayId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 97)

The ID of the Direct Connect gateway.

Type: String

Required: No

maxResults (p. 97)

The maximum number of associations to return per page.

Type: Integer

Required: No

nextToken (p. 97)

The token provided in the previous call to retrieve the next page.

Type: String

Required: No

virtualGatewayId (p. 97)

The ID of the virtual private gateway.

Type: String

Required: No

Response Syntax

```
{
```

```
   "directConnectGatewayAssociations": [
      {
         "associationState": "string",
         "directConnectGatewayId": "string",
         "stateChangeError": "string",
         "virtualGatewayId": "string",
         "virtualGatewayOwnerAccount": "string",
         "virtualGatewayRegion": "string"
      }
   ],
   "nextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

directConnectGatewayAssociations (p. 97)

The associations.

Type: Array of DirectConnectGatewayAssociation (p. 149) objects

nextToken (p. 97)

The token to retrieve the next page.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript

- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeDirectConnectGatewayAttachments

Lists the attachments between your Direct Connect gateways and virtual interfaces. You must specify a Direct Connect gateway, a virtual interface, or both. If you specify a Direct Connect gateway, the response contains all virtual interfaces attached to the Direct Connect gateway. If you specify a virtual interface, the response contains all Direct Connect gateways attached to the virtual interface. If you specify both, the response contains the attachment between the Direct Connect gateway and the virtual interface.

Request Syntax

```
{
   "directConnectGatewayId": "string",
   "maxResults": number,
   "nextToken": "string",
   "virtualInterfaceId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 100)

The ID of the Direct Connect gateway.

Type: String

Required: No

maxResults (p. 100)

The maximum number of attachments to return per page.

Type: Integer

Required: No

nextToken (p. 100)

The token provided in the previous call to retrieve the next page.

Type: String

Required: No

virtualInterfaceId (p. 100)

The ID of the virtual interface.

Type: String

Required: No

Response Syntax

```
{
```

```
   "directConnectGatewayAttachments": [
      {
         "attachmentState": "string",
         "directConnectGatewayId": "string",
         "stateChangeError": "string",
         "virtualInterfaceId": "string",
         "virtualInterfaceOwnerAccount": "string",
         "virtualInterfaceRegion": "string"
      }
   ],
   "nextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

directConnectGatewayAttachments (p. 100)

The attachments.

Type: Array of DirectConnectGatewayAttachment (p. 151) objects

nextToken (p. 100)

The token to retrieve the next page.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript

- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeDirectConnectGateways

Lists all your Direct Connect gateways or only the specified Direct Connect gateway. Deleted Direct Connect gateways are not returned.

Request Syntax

```
{
   "directConnectGatewayId": "string",
   "maxResults": number,
   "nextToken": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 103)

The ID of the Direct Connect gateway.

Type: String

Required: No

maxResults (p. 103)

The maximum number of Direct Connect gateways to return per page.

Type: Integer

Required: No

nextToken (p. 103)

The token provided in the previous call to retrieve the next page.

Type: String

Required: No

Response Syntax

```
{
   "directConnectGateways": [
      {
         "amazonSideAsn": number,
         "directConnectGatewayId": "string",
         "directConnectGatewayName": "string",
         "directConnectGatewayState": "string",
         "ownerAccount": "string",
         "stateChangeError": "string"
      }
   ],
```

```
   "nextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

directConnectGateways (p. 103)

The Direct Connect gateways.

Type: Array of DirectConnectGateway (p. 147) objects

nextToken (p. 103)

The token to retrieve the next page.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeHostedConnections

Lists the hosted connections that have been provisioned on the specified interconnect or link aggregation group (LAG).

Note
Intended for use by AWS Direct Connect partners only.

Request Syntax

```
{
   "connectionId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 105)

The ID of the interconnect or LAG.

Type: String

Required: Yes

Response Syntax

```
{
   "connections": [
      {
         "awsDevice": "string",
         "awsDeviceV2": "string",
         "bandwidth": "string",
         "connectionId": "string",
         "connectionName": "string",
         "connectionState": "string",
         "hasLogicalRedundancy": "string",
         "jumboFrameCapable": boolean,
         "lagId": "string",
         "loaIssueTime": number,
         "location": "string",
         "ownerAccount": "string",
         "partnerName": "string",
         "region": "string",
         "vlan": number
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

connections (p. 105)

The connections.

Type: Array of Connection (p. 144) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeInterconnectLoa

Deprecated. Use DescribeLoa (p. 113) instead.

Gets the LOA-CFA for the specified interconnect.

The Letter of Authorization - Connecting Facility Assignment (LOA-CFA) is a document that is used when establishing your cross connect to AWS at the colocation facility. For more information, see Requesting Cross Connects at AWS Direct Connect Locations in the *AWS Direct Connect User Guide*.

Request Syntax

```
{
   "interconnectId": "string",
   "loaContentType": "string",
   "providerName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

interconnectId (p. 107)

The ID of the interconnect.

Type: String

Required: Yes

loaContentType (p. 107)

The standard media type for the LOA-CFA document. The only supported value is application/pdf.

Type: String

Valid Values: `application/pdf`

Required: No

providerName (p. 107)

The name of the service provider who establishes connectivity on your behalf. If you supply this parameter, the LOA-CFA lists the provider name alongside your company name as the requester of the cross connect.

Type: String

Required: No

Response Syntax

```
{
   "loa": {
```

```
      "loaContent": blob,
      "loaContentType": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

loa (p. 107)

The Letter of Authorization - Connecting Facility Assignment (LOA-CFA).

Type: Loa (p. 158) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeInterconnects

Lists the interconnects owned by the AWS account or only the specified interconnect.

Request Syntax

```
{
   "interconnectId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

interconnectId (p. 109)

The ID of the interconnect.

Type: String

Required: No

Response Syntax

```
{
   "interconnects": [
      {
         "awsDevice": "string",
         "awsDeviceV2": "string",
         "bandwidth": "string",
         "hasLogicalRedundancy": "string",
         "interconnectId": "string",
         "interconnectName": "string",
         "interconnectState": "string",
         "jumboFrameCapable": boolean,
         "lagId": "string",
         "loaIssueTime": number,
         "location": "string",
         "region": "string"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

interconnects (p. 109)

The interconnects.

Type: Array of Interconnect (p. 153) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeLags

Describes all your link aggregation groups (LAG) or the specified LAG.

Request Syntax

```
{
   "lagId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

lagId (p. 111)

The ID of the LAG.

Type: String

Required: No

Response Syntax

```
{
   "lags": [
      {
         "allowsHostedConnections": boolean,
         "awsDevice": "string",
         "awsDeviceV2": "string",
         "connections": [
            {
               "awsDevice": "string",
               "awsDeviceV2": "string",
               "bandwidth": "string",
               "connectionId": "string",
               "connectionName": "string",
               "connectionState": "string",
               "hasLogicalRedundancy": "string",
               "jumboFrameCapable": boolean,
               "lagId": "string",
               "loaIssueTime": number,
               "location": "string",
               "ownerAccount": "string",
               "partnerName": "string",
               "region": "string",
               "vlan": number
            }
         ],
         "connectionsBandwidth": "string",
         "hasLogicalRedundancy": "string",
         "jumboFrameCapable": boolean,
         "lagId": "string",
         "lagName": "string",
```

```
        "lagState": "string",
        "location": "string",
        "minimumLinks": number,
        "numberOfConnections": number,
        "ownerAccount": "string",
        "region": "string"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

lags (p. 111)

The LAGs.

Type: Array of Lag (p. 155) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeLoa

Gets the LOA-CFA for a connection, interconnect, or link aggregation group (LAG).

The Letter of Authorization - Connecting Facility Assignment (LOA-CFA) is a document that is used when establishing your cross connect to AWS at the colocation facility. For more information, see Requesting Cross Connects at AWS Direct Connect Locations in the *AWS Direct Connect User Guide*.

Request Syntax

```
{
   "connectionId": "string",
   "loaContentType": "string",
   "providerName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 113)

The ID of a connection, LAG, or interconnect.

Type: String

Required: Yes

loaContentType (p. 113)

The standard media type for the LOA-CFA document. The only supported value is application/pdf.

Type: String

Valid Values: `application/pdf`

Required: No

providerName (p. 113)

The name of the service provider who establishes connectivity on your behalf. If you specify this parameter, the LOA-CFA lists the provider name alongside your company name as the requester of the cross connect.

Type: String

Required: No

Response Syntax

```
{
   "loaContent": blob,
   "loaContentType": "string"
```

```
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

loaContent (p. 113)

The binary contents of the LOA-CFA document.

Type: Base64-encoded binary data object

loaContentType (p. 113)

The standard media type for the LOA-CFA document. The only supported value is application/pdf.

Type: String

Valid Values: `application/pdf`

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeLocations

Lists the AWS Direct Connect locations in the current AWS Region. These are the locations that can be selected when calling CreateConnection (p. 45) or CreateInterconnect (p. 53).

Response Syntax

```
{
   "locations": [
      {
         "locationCode": "string",
         "locationName": "string",
         "region": "string"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

locations (p. 115)

The locations.

Type: Array of Location (p. 159) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java

- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeTags

Describes the tags associated with the specified AWS Direct Connect resources.

Request Syntax

```
{
   "resourceArns": [ "string" ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

resourceArns (p. 117)

The Amazon Resource Names (ARNs) of the resources.

Type: Array of strings

Required: Yes

Response Syntax

```
{
   "resourceTags": [
      {
         "resourceArn": "string",
         "tags": [
            {
               "key": "string",
               "value": "string"
            }
         ]
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

resourceTags (p. 117)

Information about the tags.

Type: Array of ResourceTag (p. 169) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeVirtualGateways

Lists the virtual private gateways owned by the AWS account.

You can create one or more AWS Direct Connect private virtual interfaces linked to a virtual private gateway.

Response Syntax

```
{
   "virtualGateways": [
      {
         "virtualGatewayId": "string",
         "virtualGatewayState": "string"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

virtualGateways (p. 119)

The virtual private gateways.

Type: Array of VirtualGateway (p. 172) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go

- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeVirtualInterfaces

Displays all virtual interfaces for an AWS account. Virtual interfaces deleted fewer than 15 minutes before you make the request are also returned. If you specify a connection ID, only the virtual interfaces associated with the connection are returned. If you specify a virtual interface ID, then only a single virtual interface is returned.

A virtual interface (VLAN) transmits the traffic between the AWS Direct Connect location and the customer network.

Request Syntax

```
{
   "connectionId": "string",
   "virtualInterfaceId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 121)

The ID of the connection.

Type: String

Required: No

virtualInterfaceId (p. 121)

The ID of the virtual interface.

Type: String

Required: No

Response Syntax

```
{
   "virtualInterfaces": [
      {
         "addressFamily": "string",
         "amazonAddress": "string",
         "amazonSideAsn": number,
         "asn": number,
         "authKey": "string",
         "awsDeviceV2": "string",
         "bgpPeers": [
            {
               "addressFamily": "string",
               "amazonAddress": "string",
```

```
            "asn": number,
            "authKey": "string",
            "awsDeviceV2": "string",
            "bgpPeerId": "string",
            "bgpPeerState": "string",
            "bgpStatus": "string",
            "customerAddress": "string"
         }
      ],
      "connectionId": "string",
      "customerAddress": "string",
      "customerRouterConfig": "string",
      "directConnectGatewayId": "string",
      "jumboFrameCapable": boolean,
      "location": "string",
      "mtu": number,
      "ownerAccount": "string",
      "region": "string",
      "routeFilterPrefixes": [
         {
            "cidr": "string"
         }
      ],
      "virtualGatewayId": "string",
      "virtualInterfaceId": "string",
      "virtualInterfaceName": "string",
      "virtualInterfaceState": "string",
      "virtualInterfaceType": "string",
      "vlan": number
   }
]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

virtualInterfaces (p. 121)

The virtual interfaces

Type: Array of VirtualInterface (p. 173) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DisassociateConnectionFromLag

Disassociates a connection from a link aggregation group (LAG). The connection is interrupted and re-established as a standalone connection (the connection is not deleted; to delete the connection, use the DeleteConnection (p. 75) request). If the LAG has associated virtual interfaces or hosted connections, they remain associated with the LAG. A disassociated connection owned by an AWS Direct Connect partner is automatically converted to an interconnect.

If disassociating the connection would cause the LAG to fall below its setting for minimum number of operational connections, the request fails, except when it's the last member of the LAG. If all connections are disassociated, the LAG continues to exist as an empty LAG with no physical connections.

Request Syntax

```
{
   "connectionId": "string",
   "lagId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

connectionId (p. 124)

The ID of the connection. For example, dxcon-abc123.

Type: String

Required: Yes

lagId (p. 124)

The ID of the LAG. For example, dxlag-abc123.

Type: String

Required: Yes

Response Syntax

```
{
   "awsDevice": "string",
   "awsDeviceV2": "string",
   "bandwidth": "string",
   "connectionId": "string",
   "connectionName": "string",
   "connectionState": "string",
   "hasLogicalRedundancy": "string",
   "jumboFrameCapable": boolean,
   "lagId": "string",
   "loaIssueTime": number,
   "location": "string",
```

```
   "ownerAccount": "string",
   "partnerName": "string",
   "region": "string",
   "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

awsDevice (p. 124)

This parameter has been deprecated.

The Direct Connect endpoint on which the physical connection terminates.

Type: String

awsDeviceV2 (p. 124)

The Direct Connect endpoint on which the physical connection terminates.

Type: String

bandwidth (p. 124)

The bandwidth of the connection.

Type: String

connectionId (p. 124)

The ID of the connection.

Type: String

connectionName (p. 124)

The name of the connection.

Type: String

connectionState (p. 124)

The state of the connection. The following are the possible values:

- `ordering`: The initial state of a hosted connection provisioned on an interconnect. The connection stays in the ordering state until the owner of the hosted connection confirms or declines the connection order.
- `requested`: The initial state of a standard connection. The connection stays in the requested state until the Letter of Authorization (LOA) is sent to the customer.
- `pending`: The connection has been approved and is being initialized.
- `available`: The network link is up and the connection is ready for use.
- `down`: The network link is down.
- `deleting`: The connection is being deleted.
- `deleted`: The connection has been deleted.
- `rejected`: A hosted connection in the `ordering` state enters the `rejected` state if it is deleted by the customer.

Type: String

Valid Values: `ordering | requested | pending | available | down | deleting | deleted | rejected`

hasLogicalRedundancy (p. 124)

Indicates whether the connection supports a secondary BGP peer in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

jumboFrameCapable (p. 124)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

lagId (p. 124)

The ID of the LAG.

Type: String

loaIssueTime (p. 124)

The time of the most recent call to DescribeLoa (p. 113) for this connection.

Type: Timestamp

location (p. 124)

The location of the connection.

Type: String

ownerAccount (p. 124)

The ID of the AWS account that owns the connection.

Type: String

partnerName (p. 124)

The name of the AWS Direct Connect service provider associated with the connection.

Type: String

region (p. 124)

The AWS Region where the connection is located.

Type: String

vlan (p. 124)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

TagResource

Adds the specified tags to the specified AWS Direct Connect resource. Each resource can have a maximum of 50 tags.

Each tag consists of a key and an optional value. If a tag with the same key is already associated with the resource, this action updates its value.

Request Syntax

```
{
   "resourceArn": "string",
   "tags": [
      {
         "key": "string",
         "value": "string"
      }
   ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

resourceArn (p. 128)

The Amazon Resource Name (ARN) of the resource.

Type: String

Required: Yes

tags (p. 128)

The tags to add.

Type: Array of Tag (p. 171) objects

Array Members: Minimum number of 1 item.

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

DuplicateTagKeysException

A tag key was specified more than once.

HTTP Status Code: 400

TooManyTagsException

You have reached the limit on the number of tags that can be assigned.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

UntagResource

Removes one or more tags from the specified AWS Direct Connect resource.

Request Syntax

```
{
   "resourceArn": "string",
   "tagKeys": [ "string" ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

resourceArn (p. 130)

The Amazon Resource Name (ARN) of the resource.

Type: String

Required: Yes

tagKeys (p. 130)

The tag keys of the tags to remove.

Type: Array of strings

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `^([\p{L}\p{Z}\p{N}_.:/=+\-@]*)$`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

UpdateLag

Updates the attributes of the specified link aggregation group (LAG).

You can update the following attributes:

- The name of the LAG.
- The value for the minimum number of connections that must be operational for the LAG itself to be operational.

When you create a LAG, the default value for the minimum number of operational connections is zero (0). If you update this value and the number of operational connections falls below the specified value, the LAG automatically goes down to avoid over-utilization of the remaining connections. Adjust this value with care, as it could force the LAG down if it is set higher than the current number of operational connections.

Request Syntax

```
{
   "lagId": "string",
   "lagName": "string",
   "minimumLinks": number
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

lagId (p. 132)

The ID of the LAG.

Type: String

Required: Yes

lagName (p. 132)

The name of the LAG.

Type: String

Required: No

minimumLinks (p. 132)

The minimum number of physical connections that must be operational for the LAG itself to be operational.

Type: Integer

Required: No

Response Syntax

```
{
   "allowsHostedConnections": boolean,
   "awsDevice": "string",
   "awsDeviceV2": "string",
   "connections": [
      {
         "awsDevice": "string",
         "awsDeviceV2": "string",
         "bandwidth": "string",
         "connectionId": "string",
         "connectionName": "string",
         "connectionState": "string",
         "hasLogicalRedundancy": "string",
         "jumboFrameCapable": boolean,
         "lagId": "string",
         "loaIssueTime": number,
         "location": "string",
         "ownerAccount": "string",
         "partnerName": "string",
         "region": "string",
         "vlan": number
      }
   ],
   "connectionsBandwidth": "string",
   "hasLogicalRedundancy": "string",
   "jumboFrameCapable": boolean,
   "lagId": "string",
   "lagName": "string",
   "lagState": "string",
   "location": "string",
   "minimumLinks": number,
   "numberOfConnections": number,
   "ownerAccount": "string",
   "region": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

allowsHostedConnections (p. 133)

Indicates whether the LAG can host other connections.

Type: Boolean

awsDevice (p. 133)

This parameter has been deprecated.

The Direct Connect endpoint that hosts the LAG.

Type: String

awsDeviceV2 (p. 133)

The Direct Connect endpoint that hosts the LAG.

Type: String

connections (p. 133)

The connections bundled by the LAG.

Type: Array of Connection (p. 144) objects

connectionsBandwidth (p. 133)

The individual bandwidth of the physical connections bundled by the LAG. The possible values are 1Gbps and 10Gbps.

Type: String

hasLogicalRedundancy (p. 133)

Indicates whether the LAG supports a secondary BGP peer in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

jumboFrameCapable (p. 133)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

lagId (p. 133)

The ID of the LAG.

Type: String

lagName (p. 133)

The name of the LAG.

Type: String

lagState (p. 133)

The state of the LAG. The following are the possible values:

- `requested`: The initial state of a LAG. The LAG stays in the requested state until the Letter of Authorization (LOA) is available.
- `pending`: The LAG has been approved and is being initialized.
- `available`: The network link is established and the LAG is ready for use.
- `down`: The network link is down.
- `deleting`: The LAG is being deleted.
- `deleted`: The LAG is deleted.

Type: String

Valid Values: `requested | pending | available | down | deleting | deleted`

location (p. 133)

The location of the LAG.

Type: String

minimumLinks (p. 133)

The minimum number of physical connections that must be operational for the LAG itself to be operational.

Type: Integer

numberOfConnections (p. 133)

The number of physical connections bundled by the LAG, up to a maximum of 10.

Type: Integer

ownerAccount (p. 133)

The ID of the AWS account that owns the LAG.

Type: String

region (p. 133)

The AWS Region where the connection is located.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

UpdateVirtualInterfaceAttributes

Updates the specified attributes of the specified virtual private interface.

Setting the MTU of a virtual interface to 9001 (jumbo frames) can cause an update to the underlying physical connection if it wasn't updated to support jumbo frames. Updating the connection disrupts network connectivity for all virtual interfaces associated with the connection for up to 30 seconds. To check whether your connection supports jumbo frames, call DescribeConnections (p. 93). To check whether your virtual interface supports jumbo frames, call DescribeVirtualInterfaces (p. 121).

Request Syntax

```
{
   "mtu": number,
   "virtualInterfaceId": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 177).

The request accepts the following data in JSON format.

mtu (p. 136)

The maximum transmission unit (MTU), in bytes. The supported values are 1500 and 9001. The default value is 1500.

Type: Integer

Required: No

virtualInterfaceId (p. 136)

The ID of the virtual private interface.

Type: String

Required: Yes

Response Syntax

```
{
   "addressFamily": "string",
   "amazonAddress": "string",
   "amazonSideAsn": number,
   "asn": number,
   "authKey": "string",
   "awsDeviceV2": "string",
   "bgpPeers": [
      {
         "addressFamily": "string",
         "amazonAddress": "string",
         "asn": number,
         "authKey": "string",
         "awsDeviceV2": "string",
```

```
            "bgpPeerId": "string",
            "bgpPeerState": "string",
            "bgpStatus": "string",
            "customerAddress": "string"
         }
   ],
   "connectionId": "string",
   "customerAddress": "string",
   "customerRouterConfig": "string",
   "directConnectGatewayId": "string",
   "jumboFrameCapable": boolean,
   "location": "string",
   "mtu": number,
   "ownerAccount": "string",
   "region": "string",
   "routeFilterPrefixes": [
      {
         "cidr": "string"
      }
   ],
   "virtualGatewayId": "string",
   "virtualInterfaceId": "string",
   "virtualInterfaceName": "string",
   "virtualInterfaceState": "string",
   "virtualInterfaceType": "string",
   "vlan": number
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

addressFamily (p. 136)

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

amazonAddress (p. 136)

The IP address assigned to the Amazon interface.

Type: String

amazonSideAsn (p. 136)

The autonomous system number (ASN) for the Amazon side of the connection.

Type: Long

asn (p. 136)

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

authKey (p. 136)

The authentication key for BGP configuration.

Type: String

awsDeviceV2 (p. 136)

The Direct Connect endpoint on which the virtual interface terminates.

Type: String

bgpPeers (p. 136)

The BGP peers configured on this virtual interface.

Type: Array of BGPPeer (p. 142) objects

connectionId (p. 136)

The ID of the connection.

Type: String

customerAddress (p. 136)

The IP address assigned to the customer interface.

Type: String

customerRouterConfig (p. 136)

The customer router configuration.

Type: String

directConnectGatewayId (p. 136)

The ID of the Direct Connect gateway.

Type: String

jumboFrameCapable (p. 136)

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

location (p. 136)

The location of the connection.

Type: String

mtu (p. 136)

The maximum transmission unit (MTU), in bytes. The supported values are 1500 and 9001. The default value is 1500.

Type: Integer

ownerAccount (p. 136)

The ID of the AWS account that owns the virtual interface.

Type: String

region (p. 136)

The AWS Region where the virtual interface is located.

Type: String

routeFilterPrefixes (p. 136)

The routes to be advertised to the AWS network in this Region. Applies to public virtual interfaces.

Type: Array of RouteFilterPrefix (p. 170) objects

virtualGatewayId (p. 136)

The ID of the virtual private gateway. Applies only to private virtual interfaces.

Type: String

virtualInterfaceId (p. 136)

The ID of the virtual interface.

Type: String

virtualInterfaceName (p. 136)

The name of the virtual interface assigned by the customer network.

Type: String

virtualInterfaceState (p. 136)

The state of the virtual interface. The following are the possible values:

- `confirming`: The creation of the virtual interface is pending confirmation from the virtual interface owner. If the owner of the virtual interface is different from the owner of the connection on which it is provisioned, then the virtual interface will remain in this state until it is confirmed by the virtual interface owner.
- `verifying`: This state only applies to public virtual interfaces. Each public virtual interface needs validation before the virtual interface can be created.
- `pending`: A virtual interface is in this state from the time that it is created until the virtual interface is ready to forward traffic.
- `available`: A virtual interface that is able to forward traffic.
- `down`: A virtual interface that is BGP down.
- `deleting`: A virtual interface is in this state immediately after calling DeleteVirtualInterface (p. 89) until it can no longer forward traffic.
- `deleted`: A virtual interface that cannot forward traffic.
- `rejected`: The virtual interface owner has declined creation of the virtual interface. If a virtual interface in the `Confirming` state is deleted by the virtual interface owner, the virtual interface enters the `Rejected` state.

Type: String

Valid Values: `confirming | verifying | pending | available | down | deleting | deleted | rejected`

virtualInterfaceType (p. 136)

The type of virtual interface. The possible values are `private` and `public`.

Type: String

vlan (p. 136)

The ID of the VLAN.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 179).

DirectConnectClientException

One or more parameters are not valid.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

Data Types

The AWS Direct Connect API contains several data types that various actions use. This section describes each data type in detail.

Note
The order of each element in a data type structure is not guaranteed. Applications should not assume a particular order.

The following data types are supported:

BGPPeer

Information about a BGP peer.

Contents

addressFamily

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

Required: No

amazonAddress

The IP address assigned to the Amazon interface.

Type: String

Required: No

asn

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

Required: No

authKey

The authentication key for BGP configuration.

Type: String

Required: No

awsDeviceV2

The Direct Connect endpoint on which the BGP peer terminates.

Type: String

Required: No

bgpPeerId

The ID of the BGP peer.

Type: String

Required: No

bgpPeerState

The state of the BGP peer. The following are the possible values:

- `verifying`: The BGP peering addresses or ASN require validation before the BGP peer can be created. This state applies only to public virtual interfaces.
- `pending`: The BGP peer is created, and remains in this state until it is ready to be established.
- `available`: The BGP peer is ready to be established.

- `deleting`: The BGP peer is being deleted.
- `deleted`: The BGP peer is deleted and cannot be established.

Type: String

Valid Values: `verifying | pending | available | deleting | deleted`

Required: No

bgpStatus

The status of the BGP peer. The following are the possible values:

- `up`: The BGP peer is established. This state does not indicate the state of the routing function. Ensure that you are receiving routes over the BGP session.
- `down`: The BGP peer is down.
- `unknown`: The BGP peer status is unknown.

Type: String

Valid Values: `up | down`

Required: No

customerAddress

The IP address assigned to the customer interface.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Connection

Information about an AWS Direct Connect connection.

Contents

awsDevice

This member has been deprecated.

The Direct Connect endpoint on which the physical connection terminates.

Type: String

Required: No

awsDeviceV2

The Direct Connect endpoint on which the physical connection terminates.

Type: String

Required: No

bandwidth

The bandwidth of the connection.

Type: String

Required: No

connectionId

The ID of the connection.

Type: String

Required: No

connectionName

The name of the connection.

Type: String

Required: No

connectionState

The state of the connection. The following are the possible values:

- `ordering`: The initial state of a hosted connection provisioned on an interconnect. The connection stays in the ordering state until the owner of the hosted connection confirms or declines the connection order.
- `requested`: The initial state of a standard connection. The connection stays in the requested state until the Letter of Authorization (LOA) is sent to the customer.
- `pending`: The connection has been approved and is being initialized.
- `available`: The network link is up and the connection is ready for use.
- `down`: The network link is down.
- `deleting`: The connection is being deleted.
- `deleted`: The connection has been deleted.

- `rejected`: A hosted connection in the `ordering` state enters the `rejected` state if it is deleted by the customer.

Type: String

Valid Values: `ordering | requested | pending | available | down | deleting | deleted | rejected`

Required: No

hasLogicalRedundancy

Indicates whether the connection supports a secondary BGP peer in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

Required: No

jumboFrameCapable

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

Required: No

lagId

The ID of the LAG.

Type: String

Required: No

loaIssueTime

The time of the most recent call to DescribeLoa (p. 113) for this connection.

Type: Timestamp

Required: No

location

The location of the connection.

Type: String

Required: No

ownerAccount

The ID of the AWS account that owns the connection.

Type: String

Required: No

partnerName

The name of the AWS Direct Connect service provider associated with the connection.

Type: String

Required: No

region

The AWS Region where the connection is located.

Type: String

Required: No

vlan

The ID of the VLAN.

Type: Integer

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

DirectConnectGateway

Information about a Direct Connect gateway, which enables you to connect virtual interfaces and virtual private gateways.

Contents

amazonSideAsn

The autonomous system number (ASN) for the Amazon side of the connection.

Type: Long

Required: No

directConnectGatewayId

The ID of the Direct Connect gateway.

Type: String

Required: No

directConnectGatewayName

The name of the Direct Connect gateway.

Type: String

Required: No

directConnectGatewayState

The state of the Direct Connect gateway. The following are the possible values:

- `pending`: The initial state after calling CreateDirectConnectGateway (p. 49).
- `available`: The Direct Connect gateway is ready for use.
- `deleting`: The initial state after calling DeleteDirectConnectGateway (p. 79).
- `deleted`: The Direct Connect gateway is deleted and cannot pass traffic.

Type: String

Valid Values: `pending | available | deleting | deleted`

Required: No

ownerAccount

The ID of the AWS account that owns the Direct Connect gateway.

Type: String

Required: No

stateChangeError

The error message if the state of an object failed to advance.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

DirectConnectGatewayAssociation

Information about an association between a Direct Connect gateway and a virtual private gateway.

Contents

associationState

The state of the association. The following are the possible values:

- `associating`: The initial state after calling CreateDirectConnectGatewayAssociation (p. 51).
- `associated`: The Direct Connect gateway and virtual private gateway are successfully associated and ready to pass traffic.
- `disassociating`: The initial state after calling DeleteDirectConnectGatewayAssociation (p. 81).
- `disassociated`: The virtual private gateway is disassociated from the Direct Connect gateway. Traffic flow between the Direct Connect gateway and virtual private gateway is stopped.

Type: String

Valid Values: `associating | associated | disassociating | disassociated`

Required: No

directConnectGatewayId

The ID of the Direct Connect gateway.

Type: String

Required: No

stateChangeError

The error message if the state of an object failed to advance.

Type: String

Required: No

virtualGatewayId

The ID of the virtual private gateway. Applies only to private virtual interfaces.

Type: String

Required: No

virtualGatewayOwnerAccount

The ID of the AWS account that owns the virtual private gateway.

Type: String

Required: No

virtualGatewayRegion

The AWS Region where the virtual private gateway is located.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

DirectConnectGatewayAttachment

Information about an attachment between a Direct Connect gateway and a virtual interface.

Contents

attachmentState

The state of the attachment. The following are the possible values:

- `attaching`: The initial state after a virtual interface is created using the Direct Connect gateway.
- `attached`: The Direct Connect gateway and virtual interface are attached and ready to pass traffic.
- `detaching`: The initial state after calling DeleteVirtualInterface (p. 89).
- `detached`: The virtual interface is detached from the Direct Connect gateway. Traffic flow between the Direct Connect gateway and virtual interface is stopped.

Type: String

Valid Values: `attaching | attached | detaching | detached`

Required: No

directConnectGatewayId

The ID of the Direct Connect gateway.

Type: String

Required: No

stateChangeError

The error message if the state of an object failed to advance.

Type: String

Required: No

virtualInterfaceId

The ID of the virtual interface.

Type: String

Required: No

virtualInterfaceOwnerAccount

The ID of the AWS account that owns the virtual interface.

Type: String

Required: No

virtualInterfaceRegion

The AWS Region where the virtual interface is located.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Interconnect

Information about an interconnect.

Contents

awsDevice

This member has been deprecated.

The Direct Connect endpoint on which the physical connection terminates.

Type: String

Required: No

awsDeviceV2

The Direct Connect endpoint on which the physical connection terminates.

Type: String

Required: No

bandwidth

The bandwidth of the connection.

Type: String

Required: No

hasLogicalRedundancy

Indicates whether the interconnect supports a secondary BGP in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

Required: No

interconnectId

The ID of the interconnect.

Type: String

Required: No

interconnectName

The name of the interconnect.

Type: String

Required: No

interconnectState

The state of the interconnect. The following are the possible values:

- `requested`: The initial state of an interconnect. The interconnect stays in the requested state until the Letter of Authorization (LOA) is sent to the customer.

- `pending`: The interconnect is approved, and is being initialized.
- `available`: The network link is up, and the interconnect is ready for use.
- `down`: The network link is down.
- `deleting`: The interconnect is being deleted.
- `deleted`: The interconnect is deleted.

Type: String

Valid Values: `requested | pending | available | down | deleting | deleted`

Required: No

jumboFrameCapable

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

Required: No

lagId

The ID of the LAG.

Type: String

Required: No

loaIssueTime

The time of the most recent call to DescribeLoa (p. 113) for this connection.

Type: Timestamp

Required: No

location

The location of the connection.

Type: String

Required: No

region

The AWS Region where the connection is located.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Lag

Information about a link aggregation group (LAG).

Contents

allowsHostedConnections

Indicates whether the LAG can host other connections.

Type: Boolean

Required: No

awsDevice

This member has been deprecated.

The Direct Connect endpoint that hosts the LAG.

Type: String

Required: No

awsDeviceV2

The Direct Connect endpoint that hosts the LAG.

Type: String

Required: No

connections

The connections bundled by the LAG.

Type: Array of Connection (p. 144) objects

Required: No

connectionsBandwidth

The individual bandwidth of the physical connections bundled by the LAG. The possible values are 1Gbps and 10Gbps.

Type: String

Required: No

hasLogicalRedundancy

Indicates whether the LAG supports a secondary BGP peer in the same address family (IPv4/IPv6).

Type: String

Valid Values: `unknown | yes | no`

Required: No

jumboFrameCapable

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

Required: No

lagId

The ID of the LAG.

Type: String

Required: No

lagName

The name of the LAG.

Type: String

Required: No

lagState

The state of the LAG. The following are the possible values:

- `requested`: The initial state of a LAG. The LAG stays in the requested state until the Letter of Authorization (LOA) is available.
- `pending`: The LAG has been approved and is being initialized.
- `available`: The network link is established and the LAG is ready for use.
- `down`: The network link is down.
- `deleting`: The LAG is being deleted.
- `deleted`: The LAG is deleted.

Type: String

Valid Values: `requested | pending | available | down | deleting | deleted`

Required: No

location

The location of the LAG.

Type: String

Required: No

minimumLinks

The minimum number of physical connections that must be operational for the LAG itself to be operational.

Type: Integer

Required: No

numberOfConnections

The number of physical connections bundled by the LAG, up to a maximum of 10.

Type: Integer

Required: No

ownerAccount

The ID of the AWS account that owns the LAG.

Type: String

Required: No

region

The AWS Region where the connection is located.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Loa

Information about a Letter of Authorization - Connecting Facility Assignment (LOA-CFA) for a connection.

Contents

loaContent

The binary contents of the LOA-CFA document.

Type: Base64-encoded binary data object

Required: No

loaContentType

The standard media type for the LOA-CFA document. The only supported value is application/pdf.

Type: String

Valid Values: `application/pdf`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Location

Information about an AWS Direct Connect location.

Contents

locationCode

The code for the location.

Type: String

Required: No

locationName

The name of the location. This includes the name of the colocation partner and the physical site of the building.

Type: String

Required: No

region

The AWS Region for the location.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

NewBGPPeer

Information about a new BGP peer.

Contents

addressFamily

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

Required: No

amazonAddress

The IP address assigned to the Amazon interface.

Type: String

Required: No

asn

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

Required: No

authKey

The authentication key for BGP configuration.

Type: String

Required: No

customerAddress

The IP address assigned to the customer interface.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

NewPrivateVirtualInterface

Information about a private virtual interface.

Contents

addressFamily

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

Required: No

amazonAddress

The IP address assigned to the Amazon interface.

Type: String

Required: No

asn

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

Required: Yes

authKey

The authentication key for BGP configuration.

Type: String

Required: No

customerAddress

The IP address assigned to the customer interface.

Type: String

Required: No

directConnectGatewayId

The ID of the Direct Connect gateway.

Type: String

Required: No

mtu

The maximum transmission unit (MTU), in bytes. The supported values are 1500 and 9001. The default value is 1500.

Type: Integer

Required: No

virtualGatewayId

The ID of the virtual private gateway.

Type: String

Required: No

virtualInterfaceName

The name of the virtual interface assigned by the customer network.

Type: String

Required: Yes

vlan

The ID of the VLAN.

Type: Integer

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

NewPrivateVirtualInterfaceAllocation

Information about a private virtual interface to be provisioned on a connection.

Contents

addressFamily

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

Required: No

amazonAddress

The IP address assigned to the Amazon interface.

Type: String

Required: No

asn

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

Required: Yes

authKey

The authentication key for BGP configuration.

Type: String

Required: No

customerAddress

The IP address assigned to the customer interface.

Type: String

Required: No

mtu

The maximum transmission unit (MTU), in bytes. The supported values are 1500 and 9001. The default value is 1500.

Type: Integer

Required: No

virtualInterfaceName

The name of the virtual interface assigned by the customer network.

Type: String

Required: Yes

vlan

The ID of the VLAN.

Type: Integer

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

NewPublicVirtualInterface

Information about a public virtual interface.

Contents

addressFamily

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

Required: No

amazonAddress

The IP address assigned to the Amazon interface.

Type: String

Required: No

asn

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

Required: Yes

authKey

The authentication key for BGP configuration.

Type: String

Required: No

customerAddress

The IP address assigned to the customer interface.

Type: String

Required: No

routeFilterPrefixes

The routes to be advertised to the AWS network in this Region. Applies to public virtual interfaces.

Type: Array of RouteFilterPrefix (p. 170) objects

Required: No

virtualInterfaceName

The name of the virtual interface assigned by the customer network.

Type: String

Required: Yes

vlan

The ID of the VLAN.

Type: Integer

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

NewPublicVirtualInterfaceAllocation

Information about a public virtual interface to be provisioned on a connection.

Contents

addressFamily

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

Required: No

amazonAddress

The IP address assigned to the Amazon interface.

Type: String

Required: No

asn

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

Required: Yes

authKey

The authentication key for BGP configuration.

Type: String

Required: No

customerAddress

The IP address assigned to the customer interface.

Type: String

Required: No

routeFilterPrefixes

The routes to be advertised to the AWS network in this Region. Applies to public virtual interfaces.

Type: Array of RouteFilterPrefix (p. 170) objects

Required: No

virtualInterfaceName

The name of the virtual interface assigned by the customer network.

Type: String

Required: Yes

vlan

The ID of the VLAN.

Type: Integer

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

ResourceTag

Information about a tag associated with an AWS Direct Connect resource.

Contents

resourceArn

The Amazon Resource Name (ARN) of the resource.

Type: String

Required: No

tags

The tags.

Type: Array of Tag (p. 171) objects

Array Members: Minimum number of 1 item.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

RouteFilterPrefix

Information about a route filter prefix that a customer can advertise through Border Gateway Protocol (BGP) over a public virtual interface.

Contents

cidr

The CIDR block for the advertised route. Separate multiple routes using commas. An IPv6 CIDR must use /64 or shorter.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Tag

Information about a tag.

Contents

key

The key.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `^([\p{L}\p{Z}\p{N}_.:/=+\-@]*)$`

Required: Yes

value

The value.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 256.

Pattern: `^([\p{L}\p{Z}\p{N}_.:/=+\-@]*)$`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

VirtualGateway

Information about a virtual private gateway for a private virtual interface.

Contents

virtualGatewayId

The ID of the virtual private gateway.

Type: String

Required: No

virtualGatewayState

The state of the virtual private gateway. The following are the possible values:

- `pending`: Initial state after creating the virtual private gateway.
- `available`: Ready for use by a private virtual interface.
- `deleting`: Initial state after deleting the virtual private gateway.
- `deleted`: The virtual private gateway is deleted. The private virtual interface is unable to send traffic over this gateway.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

VirtualInterface

Information about a virtual interface.

Contents

addressFamily

The address family for the BGP peer.

Type: String

Valid Values: `ipv4 | ipv6`

Required: No

amazonAddress

The IP address assigned to the Amazon interface.

Type: String

Required: No

amazonSideAsn

The autonomous system number (ASN) for the Amazon side of the connection.

Type: Long

Required: No

asn

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.

Type: Integer

Required: No

authKey

The authentication key for BGP configuration.

Type: String

Required: No

awsDeviceV2

The Direct Connect endpoint on which the virtual interface terminates.

Type: String

Required: No

bgpPeers

The BGP peers configured on this virtual interface.

Type: Array of BGPPeer (p. 142) objects

Required: No

connectionId

The ID of the connection.

Type: String

Required: No

customerAddress

The IP address assigned to the customer interface.

Type: String

Required: No

customerRouterConfig

The customer router configuration.

Type: String

Required: No

directConnectGatewayId

The ID of the Direct Connect gateway.

Type: String

Required: No

jumboFrameCapable

Indicates whether jumbo frames (9001 MTU) are supported.

Type: Boolean

Required: No

location

The location of the connection.

Type: String

Required: No

mtu

The maximum transmission unit (MTU), in bytes. The supported values are 1500 and 9001. The default value is 1500.

Type: Integer

Required: No

ownerAccount

The ID of the AWS account that owns the virtual interface.

Type: String

Required: No

region

The AWS Region where the virtual interface is located.

Type: String

Required: No

routeFilterPrefixes

The routes to be advertised to the AWS network in this Region. Applies to public virtual interfaces.

Type: Array of RouteFilterPrefix (p. 170) objects

Required: No

virtualGatewayId

The ID of the virtual private gateway. Applies only to private virtual interfaces.

Type: String

Required: No

virtualInterfaceId

The ID of the virtual interface.

Type: String

Required: No

virtualInterfaceName

The name of the virtual interface assigned by the customer network.

Type: String

Required: No

virtualInterfaceState

The state of the virtual interface. The following are the possible values:

- `confirming`: The creation of the virtual interface is pending confirmation from the virtual interface owner. If the owner of the virtual interface is different from the owner of the connection on which it is provisioned, then the virtual interface will remain in this state until it is confirmed by the virtual interface owner.
- `verifying`: This state only applies to public virtual interfaces. Each public virtual interface needs validation before the virtual interface can be created.
- `pending`: A virtual interface is in this state from the time that it is created until the virtual interface is ready to forward traffic.
- `available`: A virtual interface that is able to forward traffic.
- `down`: A virtual interface that is BGP down.
- `deleting`: A virtual interface is in this state immediately after calling DeleteVirtualInterface (p. 89) until it can no longer forward traffic.
- `deleted`: A virtual interface that cannot forward traffic.
- `rejected`: The virtual interface owner has declined creation of the virtual interface. If a virtual interface in the `Confirming` state is deleted by the virtual interface owner, the virtual interface enters the `Rejected` state.

Type: String

Valid Values: `confirming | verifying | pending | available | down | deleting | deleted | rejected`

Required: No

virtualInterfaceType

The type of virtual interface. The possible values are `private` and `public`.

Type: String

Required: No

vlan

The ID of the VLAN.

Type: Integer

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Common Parameters

The following list contains the parameters that all actions use for signing Signature Version 4 requests with a query string. Any action-specific parameters are listed in the topic for that action. For more information about Signature Version 4, see Signature Version 4 Signing Process in the *Amazon Web Services General Reference*.

Action

The action to be performed.

Type: string

Required: Yes

Version

The API version that the request is written for, expressed in the format YYYY-MM-DD.

Type: string

Required: Yes

X-Amz-Algorithm

The hash algorithm that you used to create the request signature.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Valid Values: `AWS4-HMAC-SHA256`

Required: Conditional

X-Amz-Credential

The credential scope value, which is a string that includes your access key, the date, the region you are targeting, the service you are requesting, and a termination string ("aws4_request"). The value is expressed in the following format: *access_key*/*YYYYMMDD*/*region*/*service*/aws4_request.

For more information, see Task 2: Create a String to Sign for Signature Version 4 in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-Date

The date that is used to create the signature. The format must be ISO 8601 basic format (YYYYMMDD'T'HHMMSS'Z'). For example, the following date time is a valid X-Amz-Date value: `20120325T120000Z`.

Condition: X-Amz-Date is optional for all requests; it can be used to override the date used for signing requests. If the Date header is specified in the ISO 8601 basic format, X-Amz-Date is

not required. When X-Amz-Date is used, it always overrides the value of the Date header. For more information, see Handling Dates in Signature Version 4 in the *Amazon Web Services General Reference*.

Type: string

Required: Conditional

X-Amz-Security-Token

The temporary security token that was obtained through a call to AWS Security Token Service (AWS STS). For a list of services that support temporary security credentials from AWS Security Token Service, go to AWS Services That Work with IAM in the *IAM User Guide*.

Condition: If you're using temporary security credentials from the AWS Security Token Service, you must include the security token.

Type: string

Required: Conditional

X-Amz-Signature

Specifies the hex-encoded signature that was calculated from the string to sign and the derived signing key.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-SignedHeaders

Specifies all the HTTP headers that were included as part of the canonical request. For more information about specifying signed headers, see Task 1: Create a Canonical Request For Signature Version 4 in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

Common Errors

This section lists the errors common to the API actions of all AWS services. For errors specific to an API action for this service, see the topic for that API action.

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

IncompleteSignature

The request signature does not conform to AWS standards.

HTTP Status Code: 400

InternalFailure

The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

InvalidAction

The action or operation requested is invalid. Verify that the action is typed correctly.

HTTP Status Code: 400

InvalidClientTokenId

The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403

InvalidParameterCombination

Parameters that must not be used together were used together.

HTTP Status Code: 400

InvalidParameterValue

An invalid or out-of-range value was supplied for the input parameter.

HTTP Status Code: 400

InvalidQueryParameter

The AWS query string is malformed or does not adhere to AWS standards.

HTTP Status Code: 400

MalformedQueryString

The query string contains a syntax error.

HTTP Status Code: 404

MissingAction

The request is missing an action or a required parameter.

HTTP Status Code: 400

MissingAuthenticationToken

The request must contain either a valid (registered) AWS access key ID or X.509 certificate.

HTTP Status Code: 403

MissingParameter

A required parameter for the specified action is not supplied.

HTTP Status Code: 400

OptInRequired

The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403

RequestExpired

The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400

ServiceUnavailable

The request has failed due to a temporary failure of the server.

HTTP Status Code: 503

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationError

The input fails to satisfy the constraints specified by an AWS service.

HTTP Status Code: 400