AWS Direct Connect
API Reference
API Version 2012-10-25
Table of Contents

Welcome ........................................................................................................................................... 1
Actions ............................................................................................................................................... 2

AllocateConnectionOnInterconnect ........................................................................................................ 4
  Request Syntax ................................................................................................................................. 4
  Request Parameters ......................................................................................................................... 4
  Response Syntax .............................................................................................................................. 5
  Response Elements .......................................................................................................................... 5
  Errors ........................................................................................................................................... 7
  See Also ........................................................................................................................................ 7

AllocateHostedConnection .................................................................................................................. 9
  Request Syntax ................................................................................................................................. 9
  Request Parameters ......................................................................................................................... 9
  Response Syntax .............................................................................................................................. 10
  Response Elements .......................................................................................................................... 10
  Errors ........................................................................................................................................... 12
  See Also ........................................................................................................................................ 12

AllocatePrivateVirtualInterface ......................................................................................................... 14
  Request Syntax ................................................................................................................................. 14
  Request Parameters ......................................................................................................................... 14
  Response Syntax .............................................................................................................................. 15
  Response Elements .......................................................................................................................... 15
  Errors ........................................................................................................................................... 18
  See Also ........................................................................................................................................ 18

AllocatePublicVirtualInterface ......................................................................................................... 19
  Request Syntax ................................................................................................................................. 19
  Request Parameters ......................................................................................................................... 19
  Response Syntax .............................................................................................................................. 20
  Response Elements .......................................................................................................................... 20
  Errors ........................................................................................................................................... 23
  See Also ........................................................................................................................................ 23

AssociateConnectionWithLag .............................................................................................................. 25
  Request Syntax ................................................................................................................................. 25
  Request Parameters ......................................................................................................................... 25
  Response Syntax .............................................................................................................................. 25
  Response Elements .......................................................................................................................... 26
  Errors ........................................................................................................................................... 28
  See Also ........................................................................................................................................ 28

AssociateHostedConnection .............................................................................................................. 29
  Request Syntax ................................................................................................................................. 29
  Request Parameters ......................................................................................................................... 29
  Response Syntax .............................................................................................................................. 29
  Response Elements .......................................................................................................................... 30
  Errors ........................................................................................................................................... 32
  See Also ........................................................................................................................................ 32

AssociateVirtualInterface .................................................................................................................. 33
  Request Syntax ................................................................................................................................. 33
  Request Parameters ......................................................................................................................... 33
  Response Syntax .............................................................................................................................. 33
  Response Elements .......................................................................................................................... 34
  Errors ........................................................................................................................................... 37
  See Also ........................................................................................................................................ 37

ConfirmConnection .......................................................................................................................... 38
  Request Syntax ................................................................................................................................. 38
  Request Parameters ......................................................................................................................... 38
Response Syntax ................................................................................................................................ 38
Response Elements .......................................................................................................................... 38
Errors ............................................................................................................................................... 39
See Also ........................................................................................................................................... 39
CreateLag ........................................................................................................................................... 61
CreateInterconnect .......................................................................................................................... 57
CreateDirectConnectGatewayAssociation ....................................................................................... 55
CreateDirectConnectGateway .......................................................................................................... 53
CreateConnection ............................................................................................................................. 48
CreateBGPPeer ................................................................................................................................. 45
ConfirmPublicVirtualInterface .......................................................................................................... 43
ConfirmPrivateVirtualInterface .......................................................................................................... 40
Response Syntax ............................................................................................................................... 58
Request Parameters ........................................................................................................................... 57
Request Syntax ................................................................................................................................. 57
Errors ............................................................................................................................................... 54
Response Elements ........................................................................................................................... 54
Response Syntax ............................................................................................................................... 53
Request Parameters ........................................................................................................................... 53
Request Syntax ................................................................................................................................. 53
Errors ............................................................................................................................................... 51
Response Elements ........................................................................................................................... 51
Response Syntax ............................................................................................................................... 50
Request Parameters ........................................................................................................................... 50
Request Syntax ................................................................................................................................. 50
Errors ............................................................................................................................................... 54
Response Elements ........................................................................................................................... 54
Response Syntax ............................................................................................................................... 53
Request Parameters ........................................................................................................................... 53
Request Syntax ................................................................................................................................. 53
Errors ............................................................................................................................................... 56
Response Elements ........................................................................................................................... 56
Response Syntax ............................................................................................................................... 52
Request Parameters ........................................................................................................................... 52
Request Syntax ................................................................................................................................. 52
Errors ............................................................................................................................................... 60
Response Elements ........................................................................................................................... 60
Response Syntax ............................................................................................................................... 61
Request Parameters ........................................................................................................................... 61
Request Syntax ................................................................................................................................. 61
Errors ............................................................................................................................................... 62
<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeleteLag</td>
<td>89</td>
</tr>
<tr>
<td>DeleteInterconnect</td>
<td>87</td>
</tr>
<tr>
<td>DeleteDirectConnectGateway</td>
<td>83</td>
</tr>
<tr>
<td>DeleteConnection</td>
<td>79</td>
</tr>
<tr>
<td>DeleteDirectConnectGatewayAssociation</td>
<td>85</td>
</tr>
<tr>
<td>DeleteInterconnect</td>
<td>87</td>
</tr>
<tr>
<td>DeleteLag</td>
<td>89</td>
</tr>
<tr>
<td>CreatePublicVirtualInterface</td>
<td>71</td>
</tr>
<tr>
<td>CreatePrivateVirtualInterface</td>
<td>66</td>
</tr>
<tr>
<td>Response Elements</td>
<td>63</td>
</tr>
<tr>
<td>Errors</td>
<td>64</td>
</tr>
<tr>
<td>See Also</td>
<td>65</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>66</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>66</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>66</td>
</tr>
<tr>
<td>Response Elements</td>
<td>67</td>
</tr>
<tr>
<td>Errors</td>
<td>70</td>
</tr>
<tr>
<td>See Also</td>
<td>70</td>
</tr>
<tr>
<td>CreatePublicVirtualInterface</td>
<td>71</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>71</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>71</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>72</td>
</tr>
<tr>
<td>Response Elements</td>
<td>72</td>
</tr>
<tr>
<td>Errors</td>
<td>75</td>
</tr>
<tr>
<td>See Also</td>
<td>75</td>
</tr>
<tr>
<td>DeleteBGPPeer</td>
<td>76</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>76</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>76</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>76</td>
</tr>
<tr>
<td>Response Elements</td>
<td>77</td>
</tr>
<tr>
<td>Errors</td>
<td>77</td>
</tr>
<tr>
<td>See Also</td>
<td>78</td>
</tr>
<tr>
<td>DeleteConnection</td>
<td>79</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>79</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>79</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>79</td>
</tr>
<tr>
<td>Response Elements</td>
<td>79</td>
</tr>
<tr>
<td>Errors</td>
<td>81</td>
</tr>
<tr>
<td>See Also</td>
<td>81</td>
</tr>
<tr>
<td>DeleteDirectConnectGateway</td>
<td>83</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>83</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>83</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>83</td>
</tr>
<tr>
<td>Response Elements</td>
<td>83</td>
</tr>
<tr>
<td>Errors</td>
<td>84</td>
</tr>
<tr>
<td>See Also</td>
<td>84</td>
</tr>
<tr>
<td>DeleteDirectConnectGatewayAssociation</td>
<td>85</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>85</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>85</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>85</td>
</tr>
<tr>
<td>Response Elements</td>
<td>86</td>
</tr>
<tr>
<td>Errors</td>
<td>86</td>
</tr>
<tr>
<td>See Also</td>
<td>86</td>
</tr>
<tr>
<td>DeleteInterconnect</td>
<td>87</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>87</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>87</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>87</td>
</tr>
<tr>
<td>Response Elements</td>
<td>87</td>
</tr>
<tr>
<td>Errors</td>
<td>88</td>
</tr>
<tr>
<td>See Also</td>
<td>88</td>
</tr>
<tr>
<td>DeleteLag</td>
<td>89</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>89</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>89</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>89</td>
</tr>
<tr>
<td>Response Elements</td>
<td>90</td>
</tr>
<tr>
<td>Operation</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>DescribeConnectionFromLag</td>
<td>132</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>132</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>132</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>132</td>
</tr>
<tr>
<td>Response Elements</td>
<td>132</td>
</tr>
<tr>
<td>Errors</td>
<td>132</td>
</tr>
<tr>
<td>See Also</td>
<td>132</td>
</tr>
<tr>
<td>DescribeInterconnects</td>
<td>116</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>116</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>116</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>116</td>
</tr>
<tr>
<td>Response Elements</td>
<td>116</td>
</tr>
<tr>
<td>Errors</td>
<td>117</td>
</tr>
<tr>
<td>See Also</td>
<td>117</td>
</tr>
<tr>
<td>DescribeLags</td>
<td>118</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>118</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>118</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>118</td>
</tr>
<tr>
<td>Response Elements</td>
<td>119</td>
</tr>
<tr>
<td>Errors</td>
<td>119</td>
</tr>
<tr>
<td>See Also</td>
<td>119</td>
</tr>
<tr>
<td>DescribeLoa</td>
<td>120</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>120</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>120</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>120</td>
</tr>
<tr>
<td>Response Elements</td>
<td>121</td>
</tr>
<tr>
<td>Errors</td>
<td>121</td>
</tr>
<tr>
<td>See Also</td>
<td>121</td>
</tr>
<tr>
<td>DescribeLocations</td>
<td>123</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>123</td>
</tr>
<tr>
<td>Response Elements</td>
<td>123</td>
</tr>
<tr>
<td>Errors</td>
<td>123</td>
</tr>
<tr>
<td>See Also</td>
<td>123</td>
</tr>
<tr>
<td>DescribeTags</td>
<td>125</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>125</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>125</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>125</td>
</tr>
<tr>
<td>Response Elements</td>
<td>125</td>
</tr>
<tr>
<td>Errors</td>
<td>126</td>
</tr>
<tr>
<td>See Also</td>
<td>126</td>
</tr>
<tr>
<td>DescribeVirtualGateways</td>
<td>127</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>127</td>
</tr>
<tr>
<td>Response Elements</td>
<td>127</td>
</tr>
<tr>
<td>Errors</td>
<td>127</td>
</tr>
<tr>
<td>See Also</td>
<td>127</td>
</tr>
<tr>
<td>DescribeVirtualInterfaces</td>
<td>129</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>129</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>129</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>129</td>
</tr>
<tr>
<td>Response Elements</td>
<td>130</td>
</tr>
<tr>
<td>Errors</td>
<td>130</td>
</tr>
<tr>
<td>See Also</td>
<td>130</td>
</tr>
<tr>
<td>DisassociateConnectionFromLag</td>
<td>132</td>
</tr>
<tr>
<td>Request Syntax</td>
<td>132</td>
</tr>
<tr>
<td>Request Parameters</td>
<td>132</td>
</tr>
<tr>
<td>Response Syntax</td>
<td>132</td>
</tr>
</tbody>
</table>

API Version 2012-10-25

vii
<table>
<thead>
<tr>
<th>Contents</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>See Also</td>
<td>169</td>
</tr>
<tr>
<td>NewPublicVirtualInterface</td>
<td>170</td>
</tr>
<tr>
<td>Contents</td>
<td>170</td>
</tr>
<tr>
<td>See Also</td>
<td>171</td>
</tr>
<tr>
<td>NewPublicVirtualInterfaceAllocation</td>
<td>172</td>
</tr>
<tr>
<td>Contents</td>
<td>172</td>
</tr>
<tr>
<td>See Also</td>
<td>173</td>
</tr>
<tr>
<td>ResourceTag</td>
<td>174</td>
</tr>
<tr>
<td>Contents</td>
<td>174</td>
</tr>
<tr>
<td>See Also</td>
<td>174</td>
</tr>
<tr>
<td>RouteFilterPrefix</td>
<td>175</td>
</tr>
<tr>
<td>Contents</td>
<td>175</td>
</tr>
<tr>
<td>See Also</td>
<td>175</td>
</tr>
<tr>
<td>Tag</td>
<td>176</td>
</tr>
<tr>
<td>Contents</td>
<td>176</td>
</tr>
<tr>
<td>See Also</td>
<td>176</td>
</tr>
<tr>
<td>VirtualGateway</td>
<td>177</td>
</tr>
<tr>
<td>Contents</td>
<td>177</td>
</tr>
<tr>
<td>See Also</td>
<td>177</td>
</tr>
<tr>
<td>VirtualInterface</td>
<td>178</td>
</tr>
<tr>
<td>Contents</td>
<td>178</td>
</tr>
<tr>
<td>See Also</td>
<td>181</td>
</tr>
<tr>
<td>Common Parameters</td>
<td>182</td>
</tr>
<tr>
<td>Common Errors</td>
<td>184</td>
</tr>
</tbody>
</table>
Welcome

This is the AWS Direct Connect API Reference. This guide provides detailed information about AWS Direct Connect actions, data types, parameters, and errors.

AWS Direct Connect makes it easy to establish a dedicated network connection from your premises to Amazon Web Services (AWS). Using AWS Direct Connect, you can establish private connectivity between AWS and your data center, office, or colocation environment, which in many cases can reduce your network costs, increase bandwidth throughput, and provide a more consistent network experience than Internet-based connections.

The AWS Direct Connect API Reference provides descriptions, syntax, and usage examples for each of the actions and data types for AWS Direct Connect. Use the following links to get started using the AWS Direct Connect API Reference:

- **Actions**: An alphabetical list of all AWS Direct Connect actions.
- **Data Types**: An alphabetical list of all AWS Direct Connect data types.
- **Common Query Parameters**: Parameters that all Query actions can use.
- **Common Errors**: Client and server errors that all actions can return.
Actions

The following actions are supported:

- AllocateConnectionOnInterconnect (p. 4)
- AllocateHostedConnection (p. 9)
- AllocatePrivateVirtualInterface (p. 14)
- AllocatePublicVirtualInterface (p. 19)
- AssociateConnectionWithLag (p. 25)
- AssociateHostedConnection (p. 29)
- AssociateVirtualInterface (p. 33)
- ConfirmConnection (p. 38)
- ConfirmPrivateVirtualInterface (p. 40)
- ConfirmPublicVirtualInterface (p. 43)
- CreateBGPPeer (p. 45)
- CreateConnection (p. 48)
- CreateDirectConnectGateway (p. 53)
- CreateDirectConnectGatewayAssociation (p. 55)
- CreateInterconnect (p. 57)
- CreateLag (p. 61)
- CreatePrivateVirtualInterface (p. 66)
- CreatePublicVirtualInterface (p. 71)
- DeleteBGPPeer (p. 76)
- DeleteConnection (p. 79)
- DeleteDirectConnectGateway (p. 83)
- DeleteDirectConnectGatewayAssociation (p. 85)
- DeleteInterconnect (p. 87)
- DeleteLag (p. 89)
- DeleteVirtualInterface (p. 93)
- DescribeConnectionLoa (p. 95)
- DescribeConnections (p. 98)
- DescribeConnectionsOnInterconnect (p. 100)
- DescribeDirectConnectGatewayAssociations (p. 102)
- DescribeDirectConnectGatewayAttachments (p. 105)
- DescribeDirectConnectGateways (p. 108)
- DescribeHostedConnections (p. 111)
- DescribeInterconnectLoa (p. 113)
- DescribeInterconnects (p. 116)
- DescribeLags (p. 118)
- DescribeLoa (p. 120)
- DescribeLocations (p. 123)
- DescribeTags (p. 125)
- DescribeVirtualGateways (p. 127)
- DescribeVirtualInterfaces (p. 129)
• DisassociateConnectionFromLag (p. 132)
• TagResource (p. 136)
• UntagResource (p. 138)
• UpdateLag (p. 140)
AllocateConnectionOnInterconnect

Deprecated in favor of AllocateHostedConnection (p. 9).

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 given interconnect.

**Note**
This is intended for use by AWS Direct Connect partners only.

## Request Syntax

```json
{
  "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](#).

The request accepts the following data in JSON format.

**bandwidth (p. 4)**

Bandwidth of the connection.

Example: "500Mbps"

Default: None

Values: 50Mbps, 100Mbps, 200Mbps, 300Mbps, 400Mbps, or 500Mbps

Type: String

Required: Yes

**connectionName (p. 4)**

Name of the provisioned connection.

Example: "500M Connection to AWS"

Default: None

Type: String

Required: Yes

**interconnectId (p. 4)**

ID of the interconnect on which the connection will be provisioned.

Example: dxcon-456abc78
ownerAccount (p. 4)

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

Example: 123443215678

Default: None
Type: String
Required: Yes

vlan (p. 4)

The dedicated VLAN provisioned to the connection.

Example: 101

Default: None
Type: Integer
Required: Yes

Response Syntax

```json
{
    "awsDevice": "string",
    "bandwidth": "string",
    "connectionId": "string",
    "connectionName": "string",
    "connectionState": "string",
    "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)

The Direct Connection endpoint which the physical connection terminates on.

Type: String

bandwidth (p. 5)

Bandwidth of the connection.
Example: 1Gbps (for regular connections), or 500Mbps (for hosted connections)
Default: None
Type: String
connectionId (p. 5)
The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).
Example: dxcon-fg5678gh
Default: None
Type: String
connectionName (p. 5)
The name of the connection.
Example: "My Connection to AWS"
Default: None
Type: String
connectionState (p. 5)
State of the connection.
- **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 in the process of being deleted.
- **Deleted**: The connection has been deleted.
- **Rejected**: A hosted connection in the 'Ordering' state will enter the 'Rejected' state if it is deleted by the end customer.
Type: String
Valid Values: ordering | requested | pending | available | down | deleting | deleted | rejected
lagId (p. 5)
The ID of the LAG.
Example: dxlag-fg5678gh
Type: String
loaIssueTime (p. 5)
The time of the most recent call to DescribeLoa (p. 120) for this connection.
Type: Timestamp
location (p. 5)

Where the connection is located.
Example: EqSV5
Default: None
Type: String

ownerAccount (p. 5)

The AWS account that will own the new 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.
Example: us-east-1
Default: None
Type: String

vlan (p. 5)

The VLAN ID.
Example: 101
Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 an 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 given interconnect or LAG.

Note
This is intended for use by AWS Direct Connect partners only.

Request Syntax

```json
{
  "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. 182).

The request accepts the following data in JSON format.

bandwidth (p. 9)

The bandwidth of the connection.

Example: 500Mbps

Default: None

Values: 50Mbps, 100Mbps, 200Mbps, 300Mbps, 400Mbps, or 500Mbps

Type: String

Required: Yes

collectionId (p. 9)

The ID of the interconnect or LAG on which the connection will be provisioned.

Example: dxcon-456abc78 or dxlag-abc123

Default: None

Type: String

Required: Yes

collectionName (p. 9)

The name of the provisioned connection.

Example: "500M Connection to AWS"

Default: None
ownerAccount (p. 9)

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

Example: 123443215678

Default: None

Type: String

Required: Yes

vlan (p. 9)

The dedicated VLAN provisioned to the hosted connection.

Example: 101

Default: None

Type: Integer

Required: Yes

Response Syntax

```json
{
    "awsDevice": "string",
    "bandwidth": "string",
    "connectionId": "string",
    "connectionName": "string",
    "connectionState": "string",
    "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. 10)

The Direct Connection endpoint which the physical connection terminates on.

Type: String

bandwidth (p. 10)

Bandwidth of the connection.

Example: 1Gbps (for regular connections), or 500Mbps (for hosted connections)
Default: None
Type: String
**connectionId (p. 10)**

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None
Type: String
**connectionName (p. 10)**

The name of the connection.

Example: "My Connection to AWS"

Default: None
Type: String
**connectionState (p. 10)**

State of the connection.

- **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 in the process of being deleted.
- **Deleted**: The connection has been deleted.
- **Rejected**: A hosted connection in the 'Ordering' state will enter the 'Rejected' state if it is deleted by the end customer.

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

**lagId (p. 10)**

The ID of the LAG.

Example: dxlag-fg5678gh

Type: String
**loaIssueTime (p. 10)**

The time of the most recent call to DescribeLoa (p. 120) for this connection.

Type: Timestamp
**location (p. 10)**

Where the connection is located.
Example: EqSV5
Default: None
Type: String
ownerAccount (p. 10)
The AWS account that will own the new connection.
Type: String
partnerName (p. 10)
The name of the AWS Direct Connect service provider associated with the connection.
Type: String
region (p. 10)
The AWS region where the connection is located.
Example: us-east-1
Default: None
Type: String
vlan (p. 10)
The VLAN ID.
Example: 101
Type: Integer

Errors
For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException
The API was called with invalid parameters. The error message will contain additional details about the cause.
HTTP Status Code: 400

DirectConnectServerException
A server-side error occurred during the API call. The error message will contain additional details about the cause.
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++
See Also

- 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 another AWS customer.

Virtual interfaces created using this action must be confirmed by the virtual interface owner by using the ConfirmPrivateVirtualInterface (p. 40) action. Until then, the virtual interface will be in 'Confirming' state, and will not be available for handling traffic.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

connectionId (p. 14)

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

Default: None

Type: String

Required: Yes

newPrivateVirtualInterfaceAllocation (p. 14)

Detailed information for the private virtual interface to be provisioned.

Default: None

Type: NewPrivateVirtualInterfaceAllocation (p. 168) object

Required: Yes

ownerAccount (p. 14)

The AWS account that will own the new private virtual interface.

Default: None

Type: String

Required: Yes
Response Syntax

```json
{
  "addressFamily": "string",
  "amazonAddress": "string",
  "amazonSideAsn": number,
  "asn": number,
  "authKey": "string",
  "bgpPeers": [
    {
      "addressFamily": "string",
      "amazonAddress": "string",
      "asn": number,
      "authKey": "string",
      "bgpPeerState": "string",
      "bgpStatus": "string",
      "customerAddress": "string"
    }
  ],
  "connectionId": "string",
  "customerAddress": "string",
  "customerRouterConfig": "string",
  "directConnectGatewayId": "string",
  "location": "string",
  "ownerAccount": "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. 15)**

Indicates the address family for the BGP peer.
- **ipv4**: IPv4 address family
- **ipv6**: IPv6 address family

Type: String

Valid Values: ipv4 | ipv6

**amazonAddress (p. 15)**

IP address assigned to the Amazon interface.

Example: 192.168.1.1/30 or 2001:db8::1/125

Type: String
**amazonSideAsn (p. 15)**

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

Type: Long

**asn (p. 15)**

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

Example: 65000

Type: Integer

**authKey (p. 15)**

The authentication key for BGP configuration.

Example: asdf34example

Type: String

**bgpPeers (p. 15)**

A list of the BGP peers configured on this virtual interface.

Type: Array of BGPPeer (p. 145) objects

**connectionId (p. 15)**

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: String

**customerAddress (p. 15)**

IP address assigned to the customer interface.

Example: 192.168.1.2/30 or 2001:db8::2/125

Type: String

**customerRouterConfig (p. 15)**

Information for generating the customer router configuration.

Type: String

**directConnectGatewayId (p. 15)**

The ID of the direct connect gateway.

Example: "abcd1234-dcbe-5678-be23-cdef9876ab45"

Type: String

**location (p. 15)**

Where the connection is located.

Example: EqSV5
ownerAccount (p. 15)

The AWS account that will own the new virtual interface.

Type: String

routeFilterPrefixes (p. 15)

A list of routes to be advertised to the AWS network in this region (public virtual interface).

Type: Array of RouteFilterPrefix (p. 175) objects

virtualGatewayId (p. 15)

The ID of the virtual private gateway to a VPC. This only applies to private virtual interfaces.

Example: vgw-123er56

Type: String

virtualInterfaceId (p. 15)

The ID of the virtual interface.

Example: dxvif-123dfg56

Default: None

Type: String

virtualInterfaceName (p. 15)

The name of the virtual interface assigned by the customer.

Example: "My VPC"

Type: String

virtualInterfaceState (p. 15)

State of the virtual interface.

- **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. 93) 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 will enter the ‘Rejected’ state.

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

**virtualInterfaceType (p. 15)**

The type of virtual interface.

Example: private (Amazon VPC) or public (Amazon S3, Amazon DynamoDB, and so on.)

Type: String

**vlan (p. 15)**

The VLAN ID.

Example: 101

Type: Integer

## Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 a different customer.

The owner of a connection calls this function to provision a public virtual interface which will be owned by another AWS customer.

Virtual interfaces created using this function must be confirmed by the virtual interface owner by calling ConfirmPublicVirtualInterface. Until this step has been completed, the virtual interface will be in 'Confirming' state, and will not be available for handling traffic.

When creating an IPv6 public virtual interface (addressFamily is 'ipv6'), the customer and amazon address fields should be left blank to use auto-assigned IPv6 space. Custom IPv6 Addresses are currently not supported.

Request Syntax

```json
{
   "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. 182).

The request accepts the following data in JSON format.

**connectionId (p. 19)**

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

Default: None

Type: String

Required: Yes

**newPublicVirtualInterfaceAllocation (p. 19)**

Detailed information for the public virtual interface to be provisioned.

Default: None
Response Syntax

```
{
  "addressFamily": "string",
  "amazonAddress": "string",
  "amazonSideAsn": number,
  "asn": number,
  "authKey": "string",
  "bgpPeers": [
    {
      "addressFamily": "string",
      "amazonAddress": "string",
      "asn": number,
      "authKey": "string",
      "bgpPeerState": "string",
      "bgpStatus": "string",
      "customerAddress": "string"
    }
  ],
  "connectionId": "string",
  "customerAddress": "string",
  "customerRouterConfig": "string",
  "directConnectGatewayId": "string",
  "location": "string",
  "ownerAccount": "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. 20)**

Indicates the address family for the BGP peer.
**ipv4**: IPv4 address family

**ipv6**: IPv6 address family

Type: **String**

Valid Values: ipv4 | ipv6

**amazonAddress (p. 20)**

IP address assigned to the Amazon interface.

Example: 192.168.1.1/30 or 2001:db8::1/125

Type: **String**

**amazonSideAsn (p. 20)**

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

Type: **Long**

**asn (p. 20)**

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

Example: 65000

Type: **Integer**

**authKey (p. 20)**

The authentication key for BGP configuration.

Example: asdf34example

Type: **String**

**bgpPeers (p. 20)**

A list of the BGP peers configured on this virtual interface.

Type: **Array of BGPPeer (p. 145) objects**

**connectionId (p. 20)**

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: **String**

**customerAddress (p. 20)**

IP address assigned to the customer interface.

Example: 192.168.1.2/30 or 2001:db8::2/125

Type: **String**

**customerRouterConfig (p. 20)**

Information for generating the customer router configuration.

Type: **String**
**directConnectGatewayId (p. 20)**

The ID of the direct connect gateway.

Example: "abcd1234-dcbe-5678-be23-cdef9876ab45"

Type: String

**location (p. 20)**

Where the connection is located.

Example: EqSV5

Default: None

Type: String

**ownerAccount (p. 20)**

The AWS account that will own the new virtual interface.

Type: String

**routeFilterPrefixes (p. 20)**

A list of routes to be advertised to the AWS network in this region (public virtual interface).

Type: Array of RouteFilterPrefix (p. 175) objects

**virtualGatewayId (p. 20)**

The ID of the virtual private gateway to a VPC. This only applies to private virtual interfaces.

Example: vgw-123er56

Type: String

**virtualInterfaceId (p. 20)**

The ID of the virtual interface.

Example: dxvif-123dfg56

Default: None

Type: String

**virtualInterfaceName (p. 20)**

The name of the virtual interface assigned by the customer.

Example: "My VPC"

Type: String

**virtualInterfaceState (p. 20)**

State of the virtual interface.

- **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. 93) 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 will enter the 'Rejected' state.

Type: String

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

**virtualInterfaceType (p. 20)**

The type of virtual interface.

Example: private (Amazon VPC) or public (Amazon S3, Amazon DynamoDB, and so on.)

Type: String

**vlan (p. 20)**

The VLAN ID.

Example: 101

Type: Integer

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 will be 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 reassociate a connection that's currently associated with a different LAG; however, if removing the connection will 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**

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

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

**connectionId (p. 25)**

The ID of the connection.

Example: dxcon-abc123

Default: None

Type: String

Required: Yes

**lagId (p. 25)**

The ID of the LAG with which to associate the connection.

Example: dxlag-abc123

Default: None

Type: String

Required: Yes

**Response Syntax**

```json
{
}
```
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. 25)

The Direct Connection endpoint which the physical connection terminates on.

Type: String

/bandwidth (p. 25)

Bandwidth of the connection.

Example: 1Gbps (for regular connections), or 500Mbps (for hosted connections)

Default: None

Type: String

/connectionId (p. 25)

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: String

/connectionName (p. 25)

The name of the connection.

Example: "My Connection to AWS"

Default: None

Type: String

/connectionState (p. 25)

State of the connection.

- **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 in the process of being deleted.
• **Deleted**: The connection has been deleted.
• **Rejected**: A hosted connection in the 'Ordering' state will enter the 'Rejected' state if it is deleted by the end customer.

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

**lagId (p. 25)**

The ID of the LAG.

Example: dxlag-fg5678gh

Type: String

**loaIssueTime (p. 25)**

The time of the most recent call to **DescribeLoa (p. 120)** for this connection.

Type: Timestamp

**location (p. 25)**

Where the connection is located.

Example: EqSV5

Default: None

Type: String

**ownerAccount (p. 25)**

The AWS account that will own the new connection.

Type: String

**partnerName (p. 25)**

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

Type: String

**region (p. 25)**

The AWS region where the connection is located.

Example: us-east-1

Default: None

Type: String

**vlan (p. 25)**

The VLAN ID.
Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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**

This is intended for use by AWS Direct Connect partners only.

**Request Syntax**

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

**Request Parameters**

For information about the parameters that are common to all actions, see [Common Parameters](#).

The request accepts the following data in JSON format.

**connectionId (p. 29)**

The ID of the hosted connection.

Example: dxcon-abc123

Default: None

Type: String

Required: Yes

**parentConnectionId (p. 29)**

The ID of the interconnect or the LAG.

Example: dxcon-abc123 or dxlag-abc123

Default: None

Type: String

Required: Yes

**Response Syntax**

```json
{
   "awsDevice": "string",
   "bandwidth": "string",
   "connectionId": "string",
   "connectionName": "string",
   "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.

/awsDevice (p. 29)

The Direct Connection endpoint which the physical connection terminates on.

Type: String

/bandwidth (p. 29)

Bandwidth of the connection.

Example: 1Gbps (for regular connections), or 500Mbps (for hosted connections)

Default: None

Type: String

/connectionId (p. 29)

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: String

/connectionName (p. 29)

The name of the connection.

Example: "My Connection to AWS"

Default: None

Type: String

/connectionState (p. 29)

State of the connection.

- **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 in the process of being deleted.
• **Deleted**: The connection has been deleted.
• **Rejected**: A hosted connection in the 'Ordering' state will enter the 'Rejected' state if it is deleted by the end customer.

  Type: String

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

**lagId (p. 29)**

The ID of the LAG.

Example: dxlag-fg5678gh

Type: String

**loaIssueTime (p. 29)**

The time of the most recent call to DescribeLoa (p. 120) for this connection.

Type: Timestamp

**location (p. 29)**

Where the connection is located.

Example: EqSV5

Default: None

Type: String

**ownerAccount (p. 29)**

The AWS account that will own the new connection.

Type: String

**partnerName (p. 29)**

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

Type: String

**region (p. 29)**

The AWS region where the connection is located.

Example: us-east-1

Default: None

Type: String

**vlan (p. 29)**

The VLAN ID.

Example: 101

Type: Integer
Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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. 29).

In order 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 to which the virtual interface will be newly associated.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

collectionId (p. 33)

The ID of the LAG or connection with which to associate the virtual interface.

Example: dxlag-abc123 or dxcon-abc123

Default: None

Type: String

Required: Yes

virtualInterfaceId (p. 33)

The ID of the virtual interface.

Example: dxvif-123dfg56

Default: None

Type: String

Required: Yes

Response Syntax

```
{
}
```
"addressFamily": "string",
"amazonAddress": "string",
"amazonSideAsn": number,
"asn": number,
"authKey": "string",
"bgpPeers": [
  {
    "addressFamily": "string",
    "amazonAddress": "string",
    "asn": number,
    "authKey": "string",
    "bgpPeerState": "string",
    "bgpStatus": "string",
    "customerAddress": "string"
  }
],
"connectionId": "string",
"customerAddress": "string",
"customerRouterConfig": "string",
"directConnectGatewayId": "string",
"location": "string",
"ownerAccount": "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. 33)

Indicates the address family for the BGP peer.
- ipv4: IPv4 address family
- ipv6: IPv6 address family

Type: String

Valid Values: ipv4 | ipv6

amazonAddress (p. 33)

IP address assigned to the Amazon interface.

Example: 192.168.1.1/30 or 2001:db8::1/125

Type: String

amazonSideAsn (p. 33)

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

Type: Long
asn (p. 33)
The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.
Example: 65000
Type: Integer

authKey (p. 33)
The authentication key for BGP configuration.
Example: asdf34example
Type: String

bgpPeers (p. 33)
A list of the BGP peers configured on this virtual interface.
Type: Array of BGPPeer (p. 145) objects

connectionId (p. 33)
The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).
Example: dxcon-fg5678gh
Default: None
Type: String

customerAddress (p. 33)
IP address assigned to the customer interface.
Example: 192.168.1.2/30 or 2001:db8::2/125
Type: String

customerRouterConfig (p. 33)
Information for generating the customer router configuration.
Type: String

directConnectGatewayId (p. 33)
The ID of the direct connect gateway.
Example: "abcd1234-dcba-5678-be23-cdef9876ab45"
Type: String

location (p. 33)
Where the connection is located.
Example: EqSV5
Default: None
Type: String

ownerAccount (p. 33)
The AWS account that will own the new virtual interface.
Type: String

**routeFilterPrefixes (p. 33)**

A list of routes to be advertised to the AWS network in this region (public virtual interface).

Type: Array of RouteFilterPrefix (p. 175) objects

**virtualGatewayId (p. 33)**

The ID of the virtual private gateway to a VPC. This only applies to private virtual interfaces.

Example: vgw-123er56

Type: String

**virtualInterfaceId (p. 33)**

The ID of the virtual interface.

Example: dxvif-123dfg56

Default: None

Type: String

**virtualInterfaceName (p. 33)**

The name of the virtual interface assigned by the customer.

Example: "My VPC"

Type: String

**virtualInterfaceState (p. 33)**

State of the virtual interface.

- **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. 93) 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 will enter the 'Rejected' state.

Type: String

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

**virtualInterfaceType (p. 33)**

The type of virtual interface.
Example: private (Amazon VPC) or public (Amazon S3, Amazon DynamoDB, and so on.)

Type: String

vlan (p. 33)

The VLAN ID.

Example: 101

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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

Confirm the creation of a hosted connection on an interconnect.

Upon creation, the hosted connection is initially in the 'Ordering' state, and will remain in this state until the owner calls ConfirmConnection to confirm 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. 182).

The request accepts the following data in JSON format.

connectionId (p. 38)

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

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. 38)

State of the connection.

- **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 in the process of being deleted.
• **Deleted**: The connection has been deleted.
• **Rejected**: A hosted connection in the 'Ordering' state will enter the 'Rejected' state if it is deleted by the end 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. 184)](#).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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

Accept ownership of a private virtual interface created by another customer.

After the virtual interface owner calls this function, the virtual interface will be created and attached to
the given virtual private gateway or direct connect gateway, and will be available for handling traffic.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common
Parameters (p. 182).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 40)

ID of the direct connect gateway that will be attached to the virtual interface.

A direct connect gateway can be managed via the AWS Direct Connect console or the
CreateDirectConnectGateway (p. 53) action.

Default: None

Type: String

Required: No

virtualGatewayId (p. 40)

ID of the virtual private gateway that will be attached to the virtual interface.

A virtual private gateway can be managed via the Amazon Virtual Private Cloud (VPC) console or the
EC2 CreateVpnGateway action.

Default: None

Type: String

Required: No

virtualInterfaceId (p. 40)

The ID of the virtual interface.

Example: dxvif-123dfg56

Default: None

Type: String

Required: Yes
Response Syntax

```json
{
   "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. 41)**

State of the virtual interface.

- **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. 93)` 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 will enter 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. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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

Accept ownership of a public virtual interface created by another customer.

After the virtual interface owner calls this function, the specified virtual interface will be created and made available for handling traffic.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

`virtualInterfaceId (p. 43)`

- The ID of the virtual interface.
- Example: dxvif-123dfg56
- Default: None
- Type: String
- Required: Yes

Response Syntax

```json
{
    "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. 43)`

- State of the virtual interface.
  - **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. 93)` 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 will enter 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. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 new BGP peer on a specified virtual interface. The BGP peer cannot be in the same address family (IPv4/IPv6) of an existing BGP peer on the virtual interface.

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

When creating an IPv6 BGP peer, the Amazon address and customer address fields must be left blank. IPv6 addresses are automatically assigned from Amazon's 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

```json
{
    "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. 182).

The request accepts the following data in JSON format.

**newBGPPeer (p. 45)**

Detailed information for the BGP peer to be created.

Default: None

Type: NewBGPPeer (p. 164) object

Required: No

**virtualInterfaceId (p. 45)**

The ID of the virtual interface on which the BGP peer will be provisioned.

Example: dxvif-456abc78

Default: None

Type: String

Required: No
Response Syntax

```json
{
  "virtualInterface": {
    "addressFamily": "string",
    "amazonAddress": "string",
    "amazonSideAsn": number,
    "asn": number,
    "authKey": "string",
    "bgpPeers": [
      {
        "addressFamily": "string",
        "amazonAddress": "string",
        "asn": number,
        "authKey": "string",
        "bgpPeerState": "string",
        "bgpStatus": "string",
        "customerAddress": "string"
      }
    ],
    "connectionId": "string",
    "customerAddress": "string",
    "customerRouterConfig": "string",
    "directConnectGatewayId": "string",
    "location": "string",
    "ownerAccount": "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. 46)

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

Type: VirtualInterface (p. 178) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).
DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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

A connection links your internal network to an AWS Direct Connect location over a standard 1 gigabit or 10 gigabit Ethernet fiber-optic cable. One end of the cable is connected to your router, the other to an AWS Direct Connect router. An AWS Direct Connect location provides access to Amazon Web Services in the region it is associated with. You can establish connections with AWS Direct Connect locations in multiple regions, but a connection in one region does not provide connectivity to other regions.

To find the locations for your region, use DescribeLocations (p. 123).

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 will be created.

Request Syntax

```json
{
    "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. 182).

The request accepts the following data in JSON format.

**bandwidth (p. 48)**

Bandwidth of the connection.

Example: 1Gbps

Default: None

Type: String

Required: Yes

**connectionName (p. 48)**

The name of the connection.

Example: "My Connection to AWS"

Default: None

Type: String

Required: Yes

**lagId (p. 48)**

The ID of the LAG.
Example: dxlag-fg5678gh
Type: String
Required: No

**location (p. 48)**
Where the connection is located.
Example: EqSV5
Default: None
Type: String
Required: Yes

### Response Syntax

```
{
  "awsDevice": "string",
  "bandwidth": "string",
  "connectionId": "string",
  "connectionName": "string",
  "connectionState": "string",
  "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. 49)**
The Direct Connection endpoint which the physical connection terminates on.
Type: String

**bandwidth (p. 49)**
Bandwidth of the connection.
Example: 1Gbps (for regular connections), or 500Mbps (for hosted connections)
Default: None
Type: String

**connectionId (p. 49)**
The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).
Example: dxcon-fg5678gh
Default: None
Type: String

connectionName (p. 49)

The name of the connection.
Example: "My Connection to AWS"
Default: None
Type: String

connectionState (p. 49)

State of the connection.
- **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 in the process of being deleted.
- **Deleted**: The connection has been deleted.
- **Rejected**: A hosted connection in the 'Ordering' state will enter the 'Rejected' state if it is deleted by the end customer.

Type: String

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

lagId (p. 49)

The ID of the LAG.
Example: dxlag-fg5678gh
Type: String

loaIssueTime (p. 49)

The time of the most recent call to DescribeLoa (p. 120) for this connection.
Type: Timestamp

location (p. 49)

Where the connection is located.
Example: EqSV5
Default: None
Type: String

ownerAccount (p. 49)

The AWS account that will own the new connection.
Type: String

**partnerName (p. 49)**

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

Type: String

**region (p. 49)**

The AWS region where the connection is located.

Example: us-east-1

Default: None

Type: String

**vlan (p. 49)**

The VLAN ID.

Example: 101

Type: Integer

## Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 new direct connect gateway. A direct connect gateway is an intermediate object that enables you to connect a set of virtual interfaces and virtual private gateways. Direct connect gateways are global and visible in any AWS region after they are created. The virtual interfaces and virtual private gateways that are connected through a direct connect gateway can be in different 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

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

**amazonSideAsn (p. 53)**

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.

Example: 65200

Default: 64512

Type: Long

Required: No

**directConnectGatewayName (p. 53)**

The name of the direct connect gateway.

Example: "My direct connect gateway"

Default: None

Type: String

Required: Yes

Response Syntax

```json
{
    "directConnectGateway": {
        "amazonSideAsn": number,
        "directConnectGatewayId": "string",
        "directConnectGatewayName": "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. 53)

The direct connect gateway to be created.

Type: DirectConnectGateway (p. 150) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 (VGW). The VGW 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. 182).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 55)

The ID of the direct connect gateway.

Example: "abcd1234-dcba-5678-be23-cdef9876ab45"

Default: None

Type: String

Required: Yes

virtualGatewayId (p. 55)

The ID of the virtual private gateway.

Example: "vgw-abc123ef"

Default: None

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. 55)

The direct connect gateway association to be created.

Type: DirectConnectGatewayAssociation (p. 152) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 a new interconnect between a 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 AWS Direct Connect 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 AWS Direct Connect partner’s network to an AWS Direct Connect location over a standard 1 Gbps or 10 Gbps 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 will be created.

For each end customer, the AWS Direct Connect partner provisions a connection on their interconnect by calling AllocateConnectionOnInterconnect. 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 AWS Direct Connect partner.

**Note**
This is 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. 182).

The request accepts the following data in JSON format.

**bandwidth (p. 57)**

The port bandwidth

Example: 1Gbps

Default: None

Available values: 1Gbps,10Gbps

Type: String

Required: Yes

**interconnectName (p. 57)**

The name of the interconnect.
Example: "1G Interconnect to AWS"
Default: None
Type: String
Required: Yes

**lagId (p. 57)**

The ID of the LAG.
Example: dxlag-fg5678gh
Type: String
Required: No

**location (p. 57)**

Where the interconnect is located
Example: EqSV5
Default: None
Type: String
Required: Yes

---

**Response Syntax**

```
{
  "awsDevice": "string",
  "bandwidth": "string",
  "interconnectId": "string",
  "interconnectName": "string",
  "interconnectState": "string",
  "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. 58)**

The Direct Connection endpoint which the physical connection terminates on.
Type: String

**bandwidth (p. 58)**

Bandwidth of the connection.
Example: 1Gbps
Default: None
Type: String

**interconnectId (p. 58)**

The ID of the interconnect.

Example: dxcon-abc123
Type: String

**interconnectName (p. 58)**

The name of the interconnect.

Example: "1G Interconnect to AWS"
Type: String

**interconnectState (p. 58)**

State of the interconnect.

- **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 has been 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 in the process of being deleted.
- **Deleted**: The interconnect has been deleted.

Type: String

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

**lagId (p. 58)**

The ID of the LAG.

Example: dxlag-fg5678gh
Type: String

**loaIssueTime (p. 58)**

The time of the most recent call to DescribeInterconnectLoa for this Interconnect.

Type: Timestamp

**location (p. 58)**

Where the connection is located.

Example: EqSV5
Default: None
Type: String

**region (p. 58)**

The AWS region where the connection is located.

Example: us-east-1
Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 new 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 1 gigabit or 10 gigabit interfaces, allowing you to treat them as a single interface.

All connections in a LAG must use the same bandwidth (for example, 10 Gbps), 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. 182).

The request accepts the following data in JSON format.

**connectionId (p. 61)**

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

Default: None

Type: String

Required: No

**connectionsBandwidth (p. 61)**

The bandwidth of the individual physical connections bundled by the LAG.

Default: None

Available values: 1Gbps, 10Gbps
Type: String
Required: Yes

lagName (p. 61)
The name of the LAG.
Example: "3x10G LAG to AWS"
Default: None

Type: String
Required: Yes

location (p. 61)
The AWS Direct Connect location in which the LAG should be allocated.
Example: EqSV5
Default: None

Type: Integer
Required: Yes

numberOfConnections (p. 61)
The number of physical connections initially provisioned and bundled by the LAG.
Default: None

Response Syntax

```json
{
    "allowsHostedConnections": boolean,
    "awsDevice": "string",
    "connections": [
        {
            "awsDevice": "string",
            "bandwidth": "string",
            "connectionId": "string",
            "connectionName": "string",
            "connectionState": "string",
            "lagId": "string",
            "loaIssueTime": number,
            "location": "string",
            "ownerAccount": "string",
            "partnerName": "string",
            "region": "string",
            "vlan": number
        }
    ],
    "connectionsBandwidth": "string",
    "lagId": "string",
    "lagName": "string",
    "lagState": "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. 62)

Indicates whether the LAG can host other connections.

Note
This is intended for use by AWS Direct Connect partners only.

Type: Boolean

awsDevice (p. 62)

The AWS Direct Connection endpoint that hosts the LAG.

Type: String

connections (p. 62)

A list of connections bundled by this LAG.

Type: Array of Connection (p. 147) objects

connectionsBandwidth (p. 62)

The individual bandwidth of the physical connections bundled by the LAG.

Available values: 1Gbps, 10Gbps

Type: String

lagId (p. 62)

The ID of the LAG.

Example: dxlag-fg5678gh

Type: String

lagName (p. 62)

The name of the LAG.

Type: String

lagState (p. 62)

The state of the LAG.

- **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 in the process of being deleted.
• **Deleted**: The LAG has been deleted.

Type: String

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

**location (p. 62)**

Where the connection is located.

Example: EqSV5

Default: None

Type: String

**minimumLinks (p. 62)**

The minimum number of physical connections that must be operational for the LAG itself to be operational. If the number of operational connections drops below this setting, the LAG state changes to down. This value can help to ensure that a LAG is not overutilized if a significant number of its bundled connections go down.

Type: Integer

**numberOfConnections (p. 62)**

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

Type: Integer

**ownerAccount (p. 62)**

The owner of the LAG.

Type: String

**region (p. 62)**

The AWS region where the connection is located.

Example: us-east-1

Default: None

Type: String

---

**Errors**

For information about the errors that are common to all actions, see [Common Errors (p. 184)](#).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.
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 new private virtual interface. A virtual interface is the VLAN that transports AWS Direct Connect traffic. A private virtual interface supports sending traffic to a single virtual private cloud (VPC).

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

**connectionId (p. 66)**

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: String

Required: Yes

**newPrivateVirtualInterface (p. 66)**

Detailed information for the private virtual interface to be created.

Default: None

Type: NewPrivateVirtualInterface (p. 166) object

Required: Yes

Response Syntax

```json
{
    "addressFamily": "string",
    "amazonAddress": "string",
    "asn": number,
    "authKey": "string",
    "customerAddress": "string",
    "directConnectGatewayId": "string",
    "virtualGatewayId": "string",
    "virtualInterfaceName": "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. 66)**

Indicates the address family for the BGP peer.
- **ipv4**: IPv4 address family
- **ipv6**: IPv6 address family

Type: String

Valid Values: ipv4 | ipv6

**amazonAddress (p. 66)**

IP address assigned to the Amazon interface.

Example: 192.168.1.1/30 or 2001:db8::1/125

Type: String

**amazonSideAsn (p. 66)**

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

Type: Long
asn (p. 66)
The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.
Example: 65000
Type: Integer

authKey (p. 66)
The authentication key for BGP configuration.
Example: asdf34example
Type: String

bgpPeers (p. 66)
A list of the BGP peers configured on this virtual interface.
Type: Array of BGPPeer (p. 145) objects

collectionId (p. 66)
The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).
Example: dxcon-fg5678gh
Default: None
Type: String

customerAddress (p. 66)
IP address assigned to the customer interface.
Example: 192.168.1.2/30 or 2001:db8::2/125
Type: String

customerRouterConfig (p. 66)
Information for generating the customer router configuration.
Type: String

directConnectGatewayId (p. 66)
The ID of the direct connect gateway.
Example: "abcd1234-dcba-5678-be23-cdef9876ab45"
Type: String

location (p. 66)
Where the connection is located.
Example: EqSV5
Default: None
Type: String

ownerAccount (p. 66)
The AWS account that will own the new virtual interface.
Type: String

routeFilterPrefixes (p. 66)

A list of routes to be advertised to the AWS network in this region (public virtual interface).

Type: Array of RouteFilterPrefix (p. 175) objects

virtualGatewayId (p. 66)

The ID of the virtual private gateway to a VPC. This only applies to private virtual interfaces.

Example: vgw-123er56

Type: String

virtualInterfaceId (p. 66)

The ID of the virtual interface.

Example: dxvif-123dfg56

Default: None

Type: String

virtualInterfaceName (p. 66)

The name of the virtual interface assigned by the customer.

Example: "My VPC"

Type: String

virtualInterfaceState (p. 66)

State of the virtual interface.

- **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. 93) 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 will enter the 'Rejected' state.

Type: String

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

virtualInterfaceType (p. 66)

The type of virtual interface.
Example: private (Amazon VPC) or public (Amazon S3, Amazon DynamoDB, and so on.)
Type: String

`vlan (p. 66)`
The VLAN ID.
Example: 101
Type: Integer

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 184).

**DirectConnectClientException**
The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**
A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 new 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 Simple Storage Service (Amazon S3).

When creating an IPv6 public virtual interface (addressFamily is 'ipv6'), the customer and amazon address fields should be left blank to use auto-assigned IPv6 space. Custom IPv6 Addresses are currently not supported.

Request Syntax

```json
{
    "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. 182).

The request accepts the following data in JSON format.

**connectionId (p. 71)**

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: String

Required: Yes

**newPublicVirtualInterface (p. 71)**

Detailed information for the public virtual interface to be created.

Default: None

Type: NewPublicVirtualInterface (p. 170) object

Required: Yes
Response Syntax

```
{
  "addressFamily": "string",
  "amazonAddress": "string",
  "amazonSideAsn": number,
  "asn": number,
  "authKey": "string",
  "bgpPeers": [ 
    {
      "addressFamily": "string",
      "amazonAddress": "string",
      "asn": number,
      "authKey": "string",
      "bgpPeerState": "string",
      "bgpStatus": "string",
      "customerAddress": "string"
    }
  ],
  "connectionId": "string",
  "customerAddress": "string",
  "customerRouterConfig": "string",
  "directConnectGatewayId": "string",
  "location": "string",
  "ownerAccount": "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. 72)**

Indicates the address family for the BGP peer.
- **ipv4**: IPv4 address family
- **ipv6**: IPv6 address family

Type: String

Valid Values: ipv4 | ipv6

**amazonAddress (p. 72)**

IP address assigned to the Amazon interface.

Example: 192.168.1.1/30 or 2001:db8::1/125

Type: String
amazonSideAsn (p. 72)
The autonomous system number (ASN) for the Amazon side of the connection.
Type: Long

asn (p. 72)
The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.
Example: 65000
Type: Integer

authKey (p. 72)
The authentication key for BGP configuration.
Example: asdf34example
Type: String

bgpPeers (p. 72)
A list of the BGP peers configured on this virtual interface.
Type: Array of BGPPeer (p. 145) objects

customerAddress (p. 72)
IP address assigned to the customer interface.
Example: 192.168.1.2/30 or 2001:db8::2/125
Type: String

customerRouterConfig (p. 72)
Information for generating the customer router configuration.
Type: String

directConnectGatewayId (p. 72)
The ID of the direct connect gateway.
Example: "abcd1234-dcba-5678-be23-cdef9876ab45"
Type: String

location (p. 72)
Where the connection is located.
Example: EqSV5
Default: None  
Type: String

**ownerAccount (p. 72)**  
The AWS account that will own the new virtual interface.

Type: String

**routeFilterPrefixes (p. 72)**  
A list of routes to be advertised to the AWS network in this region (public virtual interface).

Type: Array of RouteFilterPrefix (p. 175) objects

**virtualGatewayId (p. 72)**  
The ID of the virtual private gateway to a VPC. This only applies to private virtual interfaces.

Example: vgw-123er56  
Type: String

**virtualInterfaceId (p. 72)**  
The ID of the virtual interface.

Example: dxvif-123dfg56  
Default: None

Type: String

**virtualInterfaceName (p. 72)**  
The name of the virtual interface assigned by the customer.

Example: "My VPC"  
Type: String

**virtualInterfaceState (p. 72)**  
State of the virtual interface.

- **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. 93) 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 will enter the 'Rejected' state.

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

virtualInterfaceType (p. 72)

The type of virtual interface.

Example: private (Amazon VPC) or public (Amazon S3, Amazon DynamoDB, and so on.)

Type: String

vlan (p. 72)

The VLAN ID.

Example: 101

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 a BGP peer on the specified virtual interface that matches the specified customer address and ASN. You cannot delete the last BGP peer from a virtual interface.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

**asn (p. 76)**

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

Example: 65000

Type: Integer

Required: No

**customerAddress (p. 76)**

IP address assigned to the customer interface.

Example: 192.168.1.2/30 or 2001:db8::2/125

Type: String

Required: No

**virtualInterfaceId (p. 76)**

The ID of the virtual interface from which the BGP peer will be deleted.

Example: dxvif-456abc78

Default: None

Type: String

Required: No

Response Syntax

```json
{
  "virtualInterface": {
    "addressFamily": "string",
    "amazonAddress": "string",
    "amazonSideAsn": 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. 76)

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

Type: VirtualInterface (p. 178) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.
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 connection.

Deleting a connection only stops the AWS Direct Connect port hour and data transfer charges. You need to cancel separately with the providers any services or charges for cross-connects or network circuits that connect you to the AWS Direct Connect location.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

connectionId (p. 79)

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: String

Required: Yes

Response Syntax

```json
{
    "awsDevice": "string",
    "bandwidth": "string",
    "connectionId": "string",
    "connectionName": "string",
    "connectionState": "string",
    "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. 79)**

The Direct Connection endpoint which the physical connection terminates on.

Type: String

**bandwidth (p. 79)**

Bandwidth of the connection.

Example: 1Gbps (for regular connections), or 500Mbps (for hosted connections)

Default: None

Type: String

**connectionId (p. 79)**

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: String

**connectionName (p. 79)**

The name of the connection.

Example: "My Connection to AWS"

Default: None

Type: String

**connectionState (p. 79)**

State of the connection.

- **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 connection is ready for use.
- **Down**: The network link is down.
- **Deleting**: The connection is in the process of being deleted.
- **Deleted**: The connection has been deleted.
- **Rejected**: A hosted connection in the 'Ordering' state will enter the 'Rejected' state if it is deleted by the end customer.

Type: String

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

**lagId (p. 79)**

The ID of the LAG.
Example: dxlag-fg5678gh
Type: String

**loaIssueTime (p. 79)**

The time of the most recent call to DescribeLoa (p. 120) for this connection.
Type: Timestamp

**location (p. 79)**

Where the connection is located.
Example: EqSV5
Default: None
Type: String

**ownerAccount (p. 79)**

The AWS account that will own the new connection.
Type: String

**partnerName (p. 79)**

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

**region (p. 79)**

The AWS region where the connection is located.
Example: us-east-1
Default: None
Type: String

**vlan (p. 79)**

The VLAN ID.
Example: 101
Type: Integer

### Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.
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 a 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

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 83)

The ID of the direct connect gateway.

Example: "abcd1234-dcbe-5678-be23-cdef9876ab45"

Default: None

Type: String

Required: Yes

Response Syntax

```json
{
    "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. 83)

The direct connect gateway to be deleted.

Type: DirectConnectGateway (p. 150) object
Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 a direct connect gateway and a virtual private gateway.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 85)

The ID of the direct connect gateway.

Example: "abcd1234-dcba-5678-be23-cdef9876ab45"

Default: None

Type: String

Required: Yes

virtualGatewayId (p. 85)

The ID of the virtual private gateway.

Example: "vgw-abc123ef"

Default: None

Type: String

Required: Yes

Response Syntax

```json
{
    "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. 85)

The direct connect gateway association to be deleted.

Type: DirectConnectGatewayAssociation (p. 152) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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**
This is intended for use by AWS Direct Connect partners only.

**Request Syntax**

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

**Request Parameters**

For information about the parameters that are common to all actions, see [Common Parameters](#).

The request accepts the following data in JSON format.

**interconnectId** *(p. 87)*

The ID of the interconnect.

Example: dxcon-abc123

Type: String

Required: Yes

**Response Syntax**

```json
{
  "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. 87)*

State of the interconnect.

- **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 has been 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 in the process of being deleted.
- **Deleted**: The interconnect has been deleted.
Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see [Common Parameters](#).

The request accepts the following data in JSON format.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example</th>
<th>Default</th>
<th>Type</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>lagId (p. 89)</td>
<td>The ID of the LAG to delete.</td>
<td>dxlag-abc123</td>
<td>None</td>
<td>String</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Response Syntax

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

allowsHostedConnections (p. 89)

Indicates whether the LAG can host other connections.

Note
This is intended for use by AWS Direct Connect partners only.

Type: Boolean

awsDevice (p. 89)

The AWS Direct Connection endpoint that hosts the LAG.

Type: String

connections (p. 89)

A list of connections bundled by this LAG.

Type: Array of Connection (p. 147) objects

connectionsBandwidth (p. 89)

The individual bandwidth of the physical connections bundled by the LAG.

Available values: 1Gbps, 10Gbps

Type: String

lagId (p. 89)

The ID of the LAG.

Example: dxlag-fg5678gh

Type: String

lagName (p. 89)

The name of the LAG.

Type: String

lagState (p. 89)

The state of the LAG.

- 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 in the process of being deleted.
• **Deleted**: The LAG has been deleted.
  
  **Type**: String
  
  **Valid Values**: requested | pending | available | down | deleting | deleted

**location (p. 89)**

Where the connection is located.

**Example**: EqSV5

**Default**: None

**Type**: String

**minimumLinks (p. 89)**

The minimum number of physical connections that must be operational for the LAG itself to be operational. If the number of operational connections drops below this setting, the LAG state changes to down. This value can help to ensure that a LAG is not overutilized if a significant number of its bundled connections go down.

**Type**: Integer

**numberOfConnections (p. 89)**

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

**Type**: Integer

**ownerAccount (p. 89)**

The owner of the LAG.

**Type**: String

**region (p. 89)**

The AWS region where the connection is located.

**Example**: us-east-1

**Default**: None

**Type**: String

---

**Errors**

For information about the errors that are common to all actions, see [Common Errors (p. 184)](#).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

**HTTP Status Code**: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

**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

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

**virtualInterfaceId (p. 93)**

The ID of the virtual interface.

Example: dxvif-123dfg56

Default: None

Type: String

Required: Yes

Response Syntax

```json
{
   "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. 93)**

State of the virtual interface.

- **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. 93)` 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 will enter 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. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 in favor of DescribeLoa (p. 120).

Returns 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. 182).

The request accepts the following data in JSON format.

collectionId (p. 95)

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: String

Required: Yes

loaContentType (p. 95)

A standard media type indicating the content type of the LOA-CFA document. Currently, the only supported value is "application/pdf".

Default: application/pdf

Type: String

Valid Values: application/pdf

Required: No

providerName (p. 95)

The name of the APN partner or 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.

Default: None
Type: String
Required: No

Response Syntax

```json
{
  "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. 96)**

A structure containing the Letter of Authorization - Connecting Facility Assignment (LOA-CFA) for a connection.

Type: Loa (p. 162) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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
See Also

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

Displays all connections in this region.

If a connection ID is provided, the call returns only that particular connection.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

connectionId (p. 98)

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: String

Required: No

Response Syntax

```json
{
  "connections": [
    {
      "awsDevice": "string",
      "bandwidth": "string",
      "connectionId": "string",
      "connectionName": "string",
      "connectionState": "string",
      "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. 98)**

A list of connections.

Type: Array of **Connection (p. 147)** objects

## Errors

For information about the errors that are common to all actions, see [Common Errors (p. 184)](#).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 in favor of DescribeHostedConnections (p. 111).

Returns a list of connections that have been provisioned on the given interconnect.

Note
This is intended for use by AWS Direct Connect partners only.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

**interconnectId** (p. 100)

ID of the interconnect on which a list of connection is provisioned.

Example: dxcon-abc123

Default: None

Type: String

Required: Yes

Response Syntax

```json
{
    "connections": [
    {
        "awsDevice": "string",
        "bandwidth": "string",
        "connectionId": "string",
        "connectionName": "string",
        "connectionState": "string",
        "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. 100)

A list of connections.

Type: Array of Connection (p. 147) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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

Returns a list of all direct connect gateway and virtual private gateway (VGW) associations. Either a direct connect gateway ID or a VGW ID must be provided in the request. If a direct connect gateway ID is provided, the response returns all VGWs associated with the direct connect gateway. If a VGW ID is provided, the response returns all direct connect gateways associated with the VGW. If both are provided, the response only returns the association that matches both the direct connect gateway and the VGW.

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. 182).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 102)

The ID of the direct connect gateway.

Example: "abcd1234-dcba-5678-be23-cdef9876ab45"

Default: None

Type: String

Required: No

maxResults (p. 102)

The maximum number of direct connect gateway associations to return per page.

Example: 15

Default: None

Type: Integer

Required: No

nextToken (p. 102)

The token provided in the previous describe result to retrieve the next page of the result.

Default: None

Type: String

Required: No

virtualGatewayId (p. 102)

The ID of the virtual private gateway.
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. 103)

Information about the direct connect gateway associations.

Type: Array of DirectConnectGatewayAssociation (p. 152) objects

nextToken (p. 103)

Token to retrieve the next page of the result.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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

Returns a list of all direct connect gateway and virtual interface (VIF) attachments. Either a direct connect gateway ID or a VIF ID must be provided in the request. If a direct connect gateway ID is provided, the response returns all VIFs attached to the direct connect gateway. If a VIF ID is provided, the response returns all direct connect gateways attached to the VIF. If both are provided, the response only returns the attachment that matches both the direct connect gateway and the VIF.

Request Syntax

```json
{
   "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. 182).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 105)

The ID of the direct connect gateway.

Example: "abcd1234-dcbe-5678-be23-cdef9876ab45"

Default: None

Type: String

Required: No

maxResults (p. 105)

The maximum number of direct connect gateway attachments to return per page.

Example: 15

Default: None

Type: Integer

Required: No

nextToken (p. 105)

The token provided in the previous describe result to retrieve the next page of the result.

Default: None

Type: String

Required: No

virtualInterfaceId (p. 105)

The ID of the virtual interface.
Example: "dxvif-abc123ef"
Default: None
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. 106)
   Information about the direct connect gateway attachments.
   Type: Array of DirectConnectGatewayAttachment (p. 154) objects

nextToken (p. 106)
   Token to retrieve the next page of the result.
   Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException
   The API was called with invalid parameters. The error message will contain additional details about
   the cause.
   HTTP Status Code: 400

DirectConnectServerException
   A server-side error occurred during the API call. The error message will contain additional details
   about the cause.
   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

Returns a list of direct connect gateways in your account. Deleted direct connect gateways are not
returned. You can provide a direct connect gateway ID in the request to return information about
the specific direct connect gateway only. Otherwise, if a direct connect gateway ID is not provided,
information about all of your direct connect gateways is returned.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common
Parameters (p. 182).

The request accepts the following data in JSON format.

directConnectGatewayId (p. 108)

The ID of the direct connect gateway.

Example: "abcd1234-dcba-5678-be23-cdef9876ab45"

Default: None

Type: String

Required: No

maxResults (p. 108)

The maximum number of direct connect gateways to return per page.

Example: 15

Default: None

Type: Integer

Required: No

nextToken (p. 108)

The token provided in the previous describe result to retrieve the next page of the result.

Default: None

Type: String

Required: No

Response Syntax

```json
{
```

API Version 2012-10-25

108
"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. 108)

Information about the direct connect gateways.

Type: Array of DirectConnectGateway (p. 150) objects

nextToken (p. 108)

Token to retrieve the next page of the result.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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
See Also

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

Returns a list of hosted connections that have been provisioned on the given interconnect or link aggregation group (LAG).

**Note**
This is intended for use by AWS Direct Connect partners only.

**Request Syntax**

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

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

**connectionId (p. 111)**

The ID of the interconnect or LAG on which the hosted connections are provisioned.

Example: dxcon-abc123 or dxlag-abc123

Default: None

Type: String

Required: Yes

**Response Syntax**

```json
{
  "connections": [
    {
      "awsDevice": "string",
      "bandwidth": "string",
      "connectionId": "string",
      "connectionName": "string",
      "connectionState": "string",
      "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. 111)**

A list of connections.

Type: Array of [Connection (p. 147)] objects

## Errors

For information about the errors that are common to all actions, see [Common Errors (p. 184)].

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 in favor of DescribeLoa (p. 120).

Returns the LOA-CFA for an 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

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

interconnectId (p. 113)

The ID of the interconnect.

Example: dxcon-abc123

Type: String

Required: Yes

loaContentType (p. 113)

A standard media type indicating the content type of the LOA-CFA document. Currently, the only supported value is "application/pdf".

Default: 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 supply this parameter, the LOA-CFA lists the provider name alongside your company name as the requester of the cross connect.

Default: None

Type: String

Required: No
Response Syntax

```json
{
   "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. 114)**

A structure containing the Letter of Authorization - Connecting Facility Assignment (LOA-CFA) for a connection.

Type: Loa (p. 162) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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

Returns a list of interconnects owned by the AWS account.
If an interconnect ID is provided, it will only return this particular interconnect.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).
The request accepts the following data in JSON format.

interconnectId (p. 116)
The ID of the interconnect.
Example: dxcon-abc123
Type: String
Required: No

Response Syntax

```
{
  "interconnects": [
    {
      "awsDevice": "string",
      "bandwidth": "string",
      "interconnectId": "string",
      "interconnectName": "string",
      "interconnectState": "string",
      "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. 116)
A list of interconnects.
Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 the link aggregation groups (LAGs) in your account.

If a LAG ID is provided, only information about the specified LAG is returned.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

lagId (p. 118)

The ID of the LAG.

Example: dxlag-abc123

Default: None

Type: String

Required: No

Response Syntax

```
{
  "lags": [
    {
      "allowsHostedConnections": boolean,
      "awsDevice": "string",
      "connections": [
        {
          "awsDevice": "string",
          "bandwidth": "string",
          "connectionId": "string",
          "connectionName": "string",
          "connectionState": "string",
          "lagId": "string",
          "loaIssueTime": number,
          "location": "string",
          "ownerAccount": "string",
          "partnerName": "string",
          "region": "string",
          "vlan": number
        }
      ],
      "connectionsBandwidth": "string",
      "lagId": "string",
      "lagName": "string",
      "lagId": "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. 118)

A list of LAGs.

Type: Array of Lag (p. 159) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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

Returns 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. 182).

The request accepts the following data in JSON format.

connectionId (p. 120)

The ID of a connection, LAG, or interconnect for which to get the LOA-CFA information.

Example: dxcon-abc123 or dxlag-abc123

Default: None

Type: String

Required: Yes

loaContentType (p. 120)

A standard media type indicating the content type of the LOA-CFA document. Currently, the only supported value is "application/pdf".

Default: application/pdf

Type: String

Valid Values: application/pdf

Required: No

providerName (p. 120)

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.

Default: None

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. 121)**

The binary contents of the LOA-CFA document.

Type: Base64-encoded binary data object

**loaContentType (p. 121)**

A standard media type indicating the content type of the LOA-CFA document. Currently, the only supported value is "application/pdf".

Default: application/pdf

Type: String

Valid Values: application/pdf

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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

Returns the list of AWS Direct Connect locations in the current AWS region. These are the locations that may be selected when calling CreateConnection (p. 48) or CreateInterconnect (p. 57).

Response Syntax

```json
{
  "locations": [
    {
      "locationCode": "string",
      "locationName": "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. 123)**
  - A list of colocation hubs where network providers have equipment. Most regions have multiple locations available.
  - Type: Array of Location (p. 163) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

- **DirectConnectClientException**
  - The API was called with invalid parameters. The error message will contain additional details about the cause.
  - HTTP Status Code: 400

- **DirectConnectServerException**
  - A server-side error occurred during the API call. The error message will contain additional details about the cause.
  - 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++
See Also

- 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 Direct Connect resources.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

**resourceArns (p. 125)**

The Amazon Resource Names (ARNs) of the Direct Connect resources.

Type: Array of strings

Required: Yes

Response Syntax

```json
{
   "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. 125)**

Information about the tags.

Type: Array of ResourceTag (p. 174) objects
Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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

Returns a list of virtual private gateways owned by the AWS account.

You can create one or more AWS Direct Connect private virtual interfaces linking to a virtual private gateway. A virtual private gateway can be managed via Amazon Virtual Private Cloud (VPC) console or the EC2 CreateVpnGateway action.

Response Syntax

```json
{
   "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. 127)**

A list of virtual private gateways.

Type: Array of VirtualGateway (p. 177) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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
See Also

- 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.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

connectionId (p. 129)

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: String

Required: No

virtualInterfaceId (p. 129)

The ID of the virtual interface.

Example: dxvif-123dfg56

Default: None

Type: String

Required: No

Response Syntax

```
{
    "virtualInterfaces": [
    {
        "addressFamily": "string",
        "amazonAddress": "string",
        "amazonSideAsn": 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. 129)**

A list of virtual interfaces.

Type: Array of VirtualInterface (p. 178) objects

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 184).

**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.
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. 79) 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 will 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. 182).

The request accepts the following data in JSON format.

**connectionId (p. 132)**

The ID of the connection to disassociate from the LAG.

Example: dxcon-abc123

Default: None

Type: String

Required: Yes

**lagId (p. 132)**

The ID of the LAG.

Example: dxlag-abc123

Default: None

Type: String

Required: Yes

Response Syntax

```
{
  "awsDevice": "string",
  "bandwidth": "string",
  "connectionId": "string",
}
```
"connectionName": "string",
"connectionState": "string",
"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. 132)**

The Direct Connection endpoint which the physical connection terminates on.

Type: String

**bandwidth (p. 132)**

Bandwidth of the connection.

Example: 1Gbps (for regular connections), or 500Mbps (for hosted connections)

Default: None

Type: String

**connectionId (p. 132)**

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: String

**connectionName (p. 132)**

The name of the connection.

Example: "My Connection to AWS"

Default: None

Type: String

**connectionState (p. 132)**

State of the connection.

- **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 in the process of being deleted.
• **Deleted**: The connection has been deleted.
• **Rejected**: A hosted connection in the ‘Ordering’ state will enter the ‘Rejected’ state if it is deleted by the end customer.

Type: String

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

**lagId (p. 132)**

The ID of the LAG.

Example: dxlag-fg5678gh

Type: String

**loaIssueTime (p. 132)**

The time of the most recent call to DescribeLoa (p. 120) for this connection.

Type: Timestamp

**location (p. 132)**

Where the connection is located.

Example: EqSV5

Default: None

Type: String

**ownerAccount (p. 132)**

The AWS account that will own the new connection.

Type: String

**partnerName (p. 132)**

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

Type: String

**region (p. 132)**

The AWS region where the connection is located.

Example: us-east-1

Default: None

Type: String

**vlan (p. 132)**

The VLAN ID.

Example: 101

Type: Integer
Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 Direct Connect resource. Each Direct Connect 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 Direct Connect 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. 182).

The request accepts the following data in JSON format.

resourceArn (p. 136)

The Amazon Resource Name (ARN) of the Direct Connect resource.


Type: String

Required: Yes
tags (p. 136)

The list of tags to add.

Type: Array of Tag (p. 176) 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. 184).
**DirectConnectClientException**

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

**DirectConnectServerException**

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 to a Direct Connect resource.

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 Direct Connect resource.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

resourceArn (p. 138)

The Amazon Resource Name (ARN) of the Direct Connect resource.

Type: String

Required: Yes

tagKeys (p. 138)

The list of tag keys to remove.

Type: Array of strings


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. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

API Version 2012-10-25
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 a 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 will automatically go down to avoid overutilization of the remaining connections. Adjusting this value should be done with care as it could force the LAG down if the value is set higher than the current number of operational connections.

Request Syntax

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

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 182).

The request accepts the following data in JSON format.

lagId (p. 140)

The ID of the LAG to update.

Example: dxlag-abc123

Default: None

Type: String

Required: Yes

lagName (p. 140)

The name for the LAG.

Example: "3x10G LAG to AWS"

Default: None

Type: String

Required: No

minimumLinks (p. 140)

The minimum number of physical connections that must be operational for the LAG itself to be operational.
Default: None

Type: Integer

Required: No

Response Syntax

```
{
  "allowsHostedConnections": boolean,
  "awsDevice": "string",
  "connections": [
    {
      "awsDevice": "string",
      "bandwidth": "string",
      "connectionId": "string",
      "connectionName": "string",
      "connectionState": "string",
      "lagId": "string",
      "loaIssueTime": number,
      "location": "string",
      "ownerAccount": "string",
      "partnerName": "string",
      "region": "string",
      "vlan": number
    }
  ],
  "connectionsBandwidth": "string",
  "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. 141)

Indicates whether the LAG can host other connections.

Note
This is intended for use by AWS Direct Connect partners only.

Type: Boolean

awsDevice (p. 141)

The AWS Direct Connection endpoint that hosts the LAG.

Type: String

connections (p. 141)

A list of connections bundled by this LAG.
Response Elements

Type: Array of Connection (p. 147) objects

connectionsBandwidth (p. 141)

The individual bandwidth of the physical connections bundled by the LAG.

Available values: 1Gbps, 10Gbps

Type: String

lagId (p. 141)

The ID of the LAG.

Example: dxlag-fg5678gh

Type: String

lagName (p. 141)

The name of the LAG.

Type: String

lagState (p. 141)

The state of the LAG.

- **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 in the process of being deleted.
- **Deleted**: The LAG has been deleted.

Type: String

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

location (p. 141)

Where the connection is located.

Example: EqSV5

Default: None

Type: String

minimumLinks (p. 141)

The minimum number of physical connections that must be operational for the LAG itself to be operational. If the number of operational connections drops below this setting, the LAG state changes to down. This value can help to ensure that a LAG is not overutilized if a significant number of its bundled connections go down.

Type: Integer

numberOfConnections (p. 141)

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

Type: Integer
ownerAccount (p. 141)

The owner of the LAG.

Type: String

region (p. 141)

The AWS region where the connection is located.

Example: us-east-1

Default: None

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 184).

DirectConnectClientException

The API was called with invalid parameters. The error message will contain additional details about the cause.

HTTP Status Code: 400

DirectConnectServerErrorException

A server-side error occurred during the API call. The error message will contain additional details about the cause.

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 (p. 145)
- Connection (p. 147)
- DirectConnectGateway (p. 150)
- DirectConnectGatewayAssociation (p. 152)
- DirectConnectGatewayAttachment (p. 154)
- Interconnect (p. 156)
- Lag (p. 159)
- Loa (p. 162)
- Location (p. 163)
- NewBGPPeer (p. 164)
- NewPrivateVirtualInterface (p. 166)
- NewPrivateVirtualInterfaceAllocation (p. 168)
- NewPublicVirtualInterface (p. 170)
- NewPublicVirtualInterfaceAllocation (p. 172)
- ResourceTag (p. 174)
- RouteFilterPrefix (p. 175)
- Tag (p. 176)
- VirtualGateway (p. 177)
- VirtualInterface (p. 178)
**BGPPeer**

A structure containing information about a BGP peer.

**Contents**

**addressFamily**

Indicates the address family for the BGP peer.
- **ipv4**: IPv4 address family
- **ipv6**: IPv6 address family

Type: String

Valid Values: `ipv4` | `ipv6`

Required: No

**amazonAddress**

IP address assigned to the Amazon interface.

Example: 192.168.1.1/30 or 2001:db8::1/125

Type: String

Required: No

**asn**

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

Example: 65000

Type: Integer

Required: No

**authKey**

The authentication key for BGP configuration.

Example: `asdf34example`

Type: String

Required: No

**bgpPeerState**

The state of the BGP peer.

- **Verifying**: The BGP peering addresses or ASN require validation before the BGP peer can be created. This state only applies to BGP peers on a public virtual interface.
- **Pending**: The BGP peer has been created, and is in this state until it is ready to be established.
- **Available**: The BGP peer can be established.
- **Deleting**: The BGP peer is in the process of being deleted.
- **Deleted**: The BGP peer has been deleted and cannot be established.

Type: String
Valid Values: verifying | pending | available | deleting | deleted

Required: No

**bgpStatus**

The Up/Down state of the BGP peer.
- **Up**: The BGP peer is established.
- **Down**: The BGP peer is down.

Type: String

Valid Values: up | down

Required: No

**customerAddress**

IP address assigned to the customer interface.

Example: 192.168.1.2/30 or 2001:db8::2/125

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

A connection represents the physical network connection between the AWS Direct Connect location and the customer.

Contents

awsDevice

The Direct Connection endpoint which the physical connection terminates on.

Type: String

Required: No

bandwidth

Bandwidth of the connection.

Example: 1Gbps (for regular connections), or 500Mbps (for hosted connections)

Default: None

Type: String

Required: No

connectionId

The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).

Example: dxcon-fg5678gh

Default: None

Type: String

Required: No

connectionName

The name of the connection.

Example: "My Connection to AWS"

Default: None

Type: String

Required: No

connectionState

State of the connection.

- **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 in the process of being deleted.
Deleted: The connection has been deleted.
Rejected: A hosted connection in the 'Ordering' state will enter the 'Rejected' state if it is deleted by the end customer.

Type: String
Valid Values: ordering | requested | pending | available | down | deleting | deleted | rejected
Required: No

lagId
The ID of the LAG.
Example: dxlag-fg5678gh
Type: String
Required: No

loaIssueTime
The time of the most recent call to DescribeLoa (p. 120) for this connection.
Type: Timestamp
Required: No

location
Where the connection is located.
Example: EqSV5
Default: None
Type: String
Required: No

ownerAccount
The AWS account that will own the new 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.
Example: us-east-1
Default: None
Type: String
Required: No

`vlan`

The VLAN ID.

Example: 101

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

A direct connect gateway is an intermediate object that 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.

Example: "abcd1234-dcbe-5678-be23-cdef9876ab45"

Type: String

Required: No
directConnectGatewayName

The name of the direct connect gateway.

Example: "My direct connect gateway"

Default: None

Type: String

Required: No
directConnectGatewayState

State of the direct connect gateway.

- **Pending**: The initial state after calling `CreateDirectConnectGateway` (p. 53).
- **Available**: The direct connect gateway is ready for use.
- **Deleting**: The initial state after calling `DeleteDirectConnectGateway` (p. 83).
- **Deleted**: The direct connect gateway is deleted and cannot pass traffic.

Type: String

Valid Values: pending | available | deleting | deleted

Required: No

ownerAccount

The AWS account ID of the owner of the direct connect gateway.

Type: String

Required: No

stateChangeError

Error message when the state of an object fails 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

The association between a direct connect gateway and virtual private gateway.

Contents

**associationState**

State of the direct connect gateway association.

- **Associating**: The initial state after calling `CreateDirectConnectGatewayAssociation` (p. 55).
- **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. 85).
- **Disassociated**: The virtual private gateway is successfully disassociated from the direct connect gateway. Traffic flow between the direct connect gateway and virtual private gateway stops.

Type: String

Valid Values: associating | associated | disassociating | disassociated

Required: No

**directConnectGatewayId**

The ID of the direct connect gateway.

Example: "abcd1234-dcbe-5678-be23-cdef9876ab45"

Type: String

Required: No

**stateChangeError**

Error message when the state of an object fails to advance.

Type: String

Required: No

**virtualGatewayId**

The ID of the virtual private gateway to a VPC. This only applies to private virtual interfaces.

Example: vgw-123er56

Type: String

Required: No

**virtualGatewayOwnerAccount**

The AWS account ID of the owner of the virtual private gateway.

Type: String

Required: No

**virtualGatewayRegion**

The region in which the virtual private gateway is located.
Example: us-east-1
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

The association between a direct connect gateway and virtual interface.

Contents

attachmentState

State of the direct connect gateway attachment.
- **Attaching**: The initial state after a virtual interface is created using the direct connect gateway.
- **Attached**: The direct connect gateway and virtual interface are successfully attached and ready to pass traffic.
- **Detaching**: The initial state after calling DeleteVirtualInterface (p. 93) on a virtual interface that is attached to a direct connect gateway.
- **Detached**: The virtual interface is successfully detached from the direct connect gateway. Traffic flow between the direct connect gateway and virtual interface stops.

Type: String

Valid Values: attaching | attached | detaching | detached

Required: No

directConnectGatewayId

The ID of the direct connect gateway.

Example: "abcd1234-dcbe-5678-be23-cdef9876ab45"

Type: String

Required: No

stateChangeError

Error message when the state of an object fails to advance.

Type: String

Required: No

virtualInterfaceId

The ID of the virtual interface.

Example: dxvif-123dfg56

Default: None

Type: String

Required: No

virtualInterfaceOwnerAccount

The AWS account ID of the owner of the virtual interface.

Type: String

Required: No
virtualInterfaceRegion

The region in which the virtual interface is located.

Example: us-east-1

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

An interconnect is a connection that can host other connections.

Like a standard AWS Direct Connect connection, an interconnect represents the physical connection between an AWS Direct Connect partner’s network and a specific Direct Connect location. An AWS Direct Connect partner who owns an interconnect can provision hosted connections on the interconnect for their end customers, thereby providing the end customers with connectivity to AWS services.

The resources of the interconnect, including bandwidth and VLAN numbers, are shared by all of the hosted connections on the interconnect, and the owner of the interconnect determines how these resources are assigned.

Contents

awsDevice

The Direct Connection endpoint which the physical connection terminates on.

Type: String
Required: No

bandwidth

Bandwidth of the connection.

Example: 1Gbps
Default: None
Type: String
Required: No

interconnectId

The ID of the interconnect.

Example: dxcon-abc123
Type: String
Required: No

interconnectName

The name of the interconnect.

Example: "1G Interconnect to AWS"
Type: String
Required: No

interconnectState

State of the interconnect.

- **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 has been approved, and is being initialized.
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

Describes a link aggregation group (LAG). A LAG is a connection that uses the Link Aggregation Control Protocol (LACP) to logically aggregate a bundle of physical connections. Like an interconnect, it can host other connections. All connections in a LAG must terminate on the same physical AWS Direct Connect endpoint, and must be the same bandwidth.

Contents

allowsHostedConnections

Indicates whether the LAG can host other connections.

Note

This is intended for use by AWS Direct Connect partners only.

Type: Boolean

Required: No

awsDevice

The AWS Direct Connection endpoint that hosts the LAG.

Type: String

Required: No

connections

A list of connections bundled by this LAG.

Type: Array of Connection (p. 147) objects

Required: No

connectionsBandwidth

The individual bandwidth of the physical connections bundled by the LAG.

Available values: 1Gbps, 10Gbps

Type: String

Required: No

lagId

The ID of the LAG.

Example: dxlag-fg5678gh

Type: String

Required: No

lagName

The name of the LAG.

Type: String

Required: No
lagState

The state of the LAG.

- **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 in the process of being deleted.
- **Deleted**: The LAG has been deleted.

Type: String

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

Required: No

location

Where the connection is located.

Example: EqSV5

Default: None

Type: String

Required: No

minimumLinks

The minimum number of physical connections that must be operational for the LAG itself to be operational. If the number of operational connections drops below this setting, the LAG state changes to **down**. This value can help to ensure that a LAG is not overutilized if a significant number of its bundled connections go down.

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 owner of the LAG.

Type: String

Required: No

region

The AWS region where the connection is located.

Example: us-east-1

Default: None
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

A structure containing the 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

A standard media type indicating the content type of the LOA-CFA document. Currently, the only supported value is "application/pdf".

Default: 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

An AWS Direct Connect location where connections and interconnects can be requested.

Contents

locationCode

The code used to indicate the AWS Direct Connect location.

Type: String

Required: No

locationName

The name of the AWS Direct Connect location. The name includes the colocation partner name and the physical site of the lit building.

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

A structure containing information about a new BGP peer.

Contents

addressFamily

Indicates the address family for the BGP peer.

- **ipv4**: IPv4 address family
- **ipv6**: IPv6 address family

Type: String

Valid Values: `ipv4` | `ipv6`

Required: No

amazonAddress

IP address assigned to the Amazon interface.

Example: `192.168.1.1/30` or `2001:db8::1/125`

Type: String

Required: No

asn

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

Example: `65000`

Type: Integer

Required: No

authKey

The authentication key for BGP configuration.

Example: `asdf34example`

Type: String

Required: No

customerAddress

IP address assigned to the customer interface.

Example: `192.168.1.2/30` or `2001:db8::2/125`

Type: String

Required: No

See Also

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

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

A structure containing information about a new private virtual interface.

Contents

addressFamily

Indicates the address family for the BGP peer.

- **ipv4**: IPv4 address family
- **ipv6**: IPv6 address family

Type: String

Valid Values: ipv4 | ipv6

Required: No

amazonAddress

IP address assigned to the Amazon interface.

Example: 192.168.1.1/30 or 2001:db8::1/125

Type: String

Required: No

asn

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

Example: 65000

Type: Integer

Required: Yes

authKey

The authentication key for BGP configuration.

Example: asdf34example

Type: String

Required: No

customerAddress

IP address assigned to the customer interface.

Example: 192.168.1.2/30 or 2001:db8::2/125

Type: String

Required: No

directConnectGatewayId

The ID of the direct connect gateway.

Example: "abcd1234-dcba-5678-be23-cdef9876ab45"
Type: String
Required: No

virtualGatewayId
The ID of the virtual private gateway to a VPC. This only applies to private virtual interfaces.
Example: vgw-123er56

Type: String
Required: No

virtualInterfaceName
The name of the virtual interface assigned by the customer.
Example: "My VPC"

Type: String
Required: Yes

vlan
The VLAN ID.
Example: 101
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

A structure containing information about a private virtual interface that will be provisioned on a connection.

Contents

addressFamily

Indicates the address family for the BGP peer.
- ipv4: IPv4 address family
- ipv6: IPv6 address family

Type: String

Valid Values: ipv4 | ipv6

Required: No

amazonAddress

IP address assigned to the Amazon interface.
Example: 192.168.1.1/30 or 2001:db8::1/125

Type: String

Required: No

asn

The autonomous system (AS) number for Border Gateway Protocol (BGP) configuration.
Example: 65000

Type: Integer

Required: Yes

authKey

The authentication key for BGP configuration.
Example: asdf34example

Type: String

Required: No

customerAddress

IP address assigned to the customer interface.
Example: 192.168.1.2/30 or 2001:db8::2/125

Type: String

Required: No

virtualInterfaceName

The name of the virtual interface assigned by the customer.
Example: "My VPC"
Type: String
Required: Yes

**vlan**

The VLAN ID.
Example: 101
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

A structure containing information about a new public virtual interface.

Contents

addressFamily

Indicates the address family for the BGP peer.

- ipv4: IPv4 address family
- ipv6: IPv6 address family

Type: String
Valid Values: ipv4 | ipv6
Required: No

amazonAddress

IP address assigned to the Amazon interface.

Example: 192.168.1.1/30 or 2001:db8::1/125

Type: String
Required: No

asn

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

Example: 65000

Type: Integer
Required: Yes

authKey

The authentication key for BGP configuration.

Example: asdf34example

Type: String
Required: No

customerAddress

IP address assigned to the customer interface.

Example: 192.168.1.2/30 or 2001:db8::2/125

Type: String
Required: No

routeFilterPrefixes

A list of routes to be advertised to the AWS network in this region (public virtual interface).

Type: Array of RouteFilterPrefix (p. 175) objects
virtualInterfaceName

The name of the virtual interface assigned by the customer.

Example: "My VPC"

Type: String

Required: Yes

vlan

The VLAN ID.

Example: 101

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

A structure containing information about a public virtual interface that will be provisioned on a connection.

Contents

addressFamily

Indicates the address family for the BGP peer.

- **ipv4**: IPv4 address family
- **ipv6**: IPv6 address family

Type: String

Valid Values: ipv4 | ipv6

Required: No

amazonAddress

IP address assigned to the Amazon interface.

Example: 192.168.1.1/30 or 2001:db8::1/125

Type: String

Required: No

asn

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

Example: 65000

Type: Integer

Required: Yes

authKey

The authentication key for BGP configuration.

Example: asdf34example

Type: String

Required: No

customerAddress

IP address assigned to the customer interface.

Example: 192.168.1.2/30 or 2001:db8::2/125

Type: String

Required: No

routeFilterPrefixes

A list of routes to be advertised to the AWS network in this region (public virtual interface).
Type: Array of RouteFilterPrefix (p. 175) objects

Required: No

**virtualInterfaceName**

The name of the virtual interface assigned by the customer.

Example: "My VPC"

Type: String

Required: Yes

**vlan**

The VLAN ID.

Example: 101

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

The tags associated with a Direct Connect resource.

Contents

resourceArn

The Amazon Resource Name (ARN) of the Direct Connect resource.

Type: String
Required: No

tags

The tags.

Type: Array of Tag (p. 176) 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

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

Contents

cidr

CIDR notation for the advertised route. Multiple routes are separated by commas.

IPv6 CIDRs must be at least a /64 or shorter

Example: 10.10.10.0/24,10.10.11.0/24,2001:db8::/64

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 of the tag.

Type: String


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

Required: Yes

value

The value of the tag.

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

You can create one or more AWS Direct Connect private virtual interfaces linking to your virtual private gateway.

Virtual private gateways can be managed using the Amazon Virtual Private Cloud (Amazon VPC) console or the Amazon EC2 CreateVpnGateway action.

Contents

virtualGatewayId

The ID of the virtual private gateway to a VPC. This only applies to private virtual interfaces.

Example: vgw-123er56

Type: String

Required: No

virtualGatewayState

State of the virtual private gateway.

- **Pending**: This is the initial state after calling `CreateVpnGateway`.
- **Available**: Ready for use by a private virtual interface.
- **Deleting**: This is the initial state after calling `DeleteVpnGateway`.
- **Deleted**: In this state, a 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

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

Contents

addressFamily

Indicates the address family for the BGP peer.
- ipv4: IPv4 address family
- ipv6: IPv6 address family

Type: String
Valid Values: ipv4 | ipv6
Required: No

amazonAddress

IP address assigned to the Amazon interface.
Example: 192.168.1.1/30 or 2001:db8::1/125

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.
Example: 65000

Type: Integer
Required: No

authKey

The authentication key for BGP configuration.
Example: asdf34example

Type: String
Required: No

bgpPeers

A list of the BGP peers configured on this virtual interface.

Type: Array of BGPPeer (p. 145) objects
connectionId
The ID of the connection. This field is also used as the ID type for operations that use multiple connection types (LAG, interconnect, and/or connection).
Example: dxcon-fg5678gh
Default: None
Type: String
Required: No

customerAddress
IP address assigned to the customer interface.
Example: 192.168.1.2/30 or 2001:db8::2/125
Type: String
Required: No

customerRouterConfig
Information for generating the customer router configuration.
Type: String
Required: No

directConnectGatewayId
The ID of the direct connect gateway.
Example: "abcd1234-dcba-5678-be23-cdef9876ab45"
Type: String
Required: No

location
Where the connection is located.
Example: EqSV5
Default: None
Type: String
Required: No

ownerAccount
The AWS account that will own the new virtual interface.
Type: String
Required: No

routeFilterPrefixes
A list of routes to be advertised to the AWS network in this region (public virtual interface).
Type: Array of RouteFilterPrefix (p. 175) objects

Required: No

virtualGatewayId

The ID of the virtual private gateway to a VPC. This only applies to private virtual interfaces.

Example: vgw-123er56

Type: String

Required: No

virtualInterfaceId

The ID of the virtual interface.

Example: dxvif-123dfg56

Default: None

Type: String

Required: No

virtualInterfaceName

The name of the virtual interface assigned by the customer.

Example: "My VPC"

Type: String

Required: No

virtualInterfaceState

State of the virtual interface.

- 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. 93) 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 will enter the 'Rejected' state.

Type: String

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

Required: No
virtualInterfaceType

The type of virtual interface.

Example: private (Amazon VPC) or public (Amazon S3, Amazon DynamoDB, and so on.)

Type: String
Required: No

vlan

The VLAN ID.

Example: 101

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'HHHMMS'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