OpenShift Dedicated 4

Cloud infrastructure access

Cloud infrastructure access with OpenShift Dedicated 4

Last Updated: 2020-06-18

OpenShift Dedicated 4 Cloud infrastructure access

Cloud infrastructure access with OpenShift Dedicated 4

Legal Notice

Abstract

Cloud infrastructure with OpenShift Dedicated 4

Table of Contents

CHAPTER 1. UNDERSTANDING CLOUD INFRASTRUCTURE ACCESS

Amazon Web Services (AWS) infrastructure access permits Customer Portal Organization Administrators and cluster owners to enable AWS Identity and Access Management (IAM) users to have federated access to the AWS Management Console for their OpenShift Dedicated cluster.

1.1. ENABLING AWS ACCESS

AWS access can be granted for customer AWS users, and private cluster access can be implemented to suit the needs of your OpenShift Dedicated environment.

Get started with Accessing AWS infrastructure for your OpenShift Dedicated cluster. By creating an AWS user and account and providing that user with access to the OpenShift Dedicated AWS account.

After you have access to the OpenShift Dedicated AWS account, use one or more of the following methods to establish a private connection to your cluster:

- Configuring AWS VPC peering: Enable VPC peering to route network traffic between two private IP addresses.
- Configuring AWS VPN: Establish a Virtual Private Network to securely connect your private network to your Amazon Virtual Private Cloud.
- Configuring AWS Direct Connect: Configure AWS Direct Connect to establish a dedicated network connection between your private network and an AWS Direct Connect location.

After configuring your cloud infrastructure access, learn more about Configuring a private cluster.

CHAPTER 2. ACCESSING AWS INFRASTRUCTURE

Amazon Web Services (AWS) infrastructure access allows Customer Portal Organization Administrators and cluster owners to enable AWS Identity and Access Management (IAM) users to have federated access to the AWS Management Console for their OpenShift Dedicated cluster. Administrators can select between Network Management or Read-only access options.

2.1. CONFIGURING AWS INFRASTRUCTURE ACCESS

Prerequisites

- An AWS account with IAM permissions.

2.1.1. Creating an AWS account with IAM permissions

Before you can configure access to AWS infrastructure, you will need to set up IAM permissions in your AWS account.

Procedure

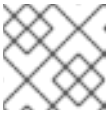
1. Log in to your AWS account. If necessary, you can create a new AWS account by following AWS documentation.
2. Create an IAM user with **STS:AllowAssumeRole** permissions within the AWS account.
   a. Open the IAM dashboard of the AWS Management Console.
   b. In the **Policies** section, click **Create Policy**.
   c. Select the **JSON** tab and replace the existing text with the following:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "*"
    }
  ]
}
```

   d. Click **Review Policy**.
   e. Provide an appropriate name and description, then click **Create Policy**.
   f. In the **Users** section, click **Add user**.
   g. Provide an appropriate user name.
   h. Select **AWS Management Console access** and other roles as needed.
   i. Adjust the password requirements as necessary for your organization, then click **Next: Policy**.

j. Click the **Attach existing policies directly** option.

k. Search for and check the policy created in previous steps.



NOTE

It is not recommended to set a permissions boundary.

l. Click **Next: Tags**, then click **Next: Review**. Confirm the configuration is correct.

m. Click **Create user**, then click **Close** on the success page.

3. Gather the IAM user's Amazon Resource Name (ARN). The ARN will have the following format: **arn:aws:iam::000111222333:user/username**.

2.1.2. Granting the IAM role from the OpenShift Cluster Manager

Procedure

1. Open the OpenShift Dedicated Cluster Manager in your browser and select the cluster you want to allow AWS infrastructure access.
2. Select the **Access control** tab, and scroll to the **AWS Infrastructure Access** section.
3. Paste the AWS IAM ARN and select **Network Management** or **Read-only** permissions, then click **Grant role**.
4. Copy the AWS OSD Console URL to your clipboard.
5. Sign in to your AWS account with your Account ID or alias, IAM user name, and password.
6. In a new browser tab, paste the AWS OSD Console URL that will be used to route to the AWS Switch Role page.
7. Your account number and role will be filled in already. Choose a display name if necessary, then click **Switch Role**. You will now see **VPC** under **Recently visited services**.

CHAPTER 3. CONFIGURING AWS VPC PEERING

This sample process configures an Amazon Web Services (AWS) VPC containing an OpenShift Dedicated cluster to peer with another AWS VPC network. For more information about creating an AWS VPC Peering connection or for other possible configurations, see the AWS VPC Peering guide.

3.1. VPC PEERING TERMS

When setting up a VPC peering connection between two VPCs on two separate AWS accounts, the following terms are used:

OSD AWS Account	The AWS account that contains the OpenShift Dedicated cluster.
OSD Cluster VPC	The VPC that contains the OpenShift Dedicated cluster.
Customer AWS Account	Your non-OSD AWS Account that you would like to peer with.
Customer VPC	The VPC in your AWS Account that you would like to peer with.
Customer VPC Region	The region where the customer's VPC resides.

NOTE

As of July 2018, AWS supports inter-region VPC peering between all commercial regions excluding China.

3.2. INITIATING THE VPC PEER REQUEST

You can send a VPC peering connection request from the OSD AWS Account to the Customer AWS Account.

Prerequisites

- Gather the following information about the Customer VPC required to initiate the peering request:
  - Customer AWS account number
  - Customer VPC ID
  - Customer VPC Region
  - Customer VPC CIDR
- Check the CIDR block used by the OpenShift Dedicated Cluster VPC. If it overlaps or matches the CIDR block for the Customer VPC, then peering between these two VPCs is not possible;

see the Amazon VPC Unsupported VPC Peering Configurations documentation for details. If the CIDR blocks do not overlap, you can procedure with the procedure.

Procedure

1. Log in to the Web Console for the OSD AWS Account and navigate to the **VPC Dashboard** in the region where the cluster is being hosted.
2. Go to the **Peering Connections** page and click the **Create Peering Connection** button.
3. Verify the details of the account you are logged in to and the details of the account and VPC you are connecting to:
    a. **Peering connection name tag** Set a descriptive name for the VPC Peering Connection.
    b. **VPC (Requester)**: Select the OpenShift Dedicated Cluster VPC ID from the dropdown *list.
    c. **Account**: Select **Another account** and provide the Customer AWS Account number * (without dashes).
    d. **Region**: If the Customer VPC Region differs from the current region, select **Another Region** and select the customer VPC Region from the dropdown list.
    e. **VPC (Accepter)**: Set the Customer VPC ID.
4. Click **Create Peering Connection**.
5. Confirm that the request enters a **Pending** state. If it enters a **Failed** state, confirm the details and repeat the process.

Additional resources

- Logging into the Web Console for the OSD AWS Account

3.3. ACCEPTING THE VPC PEER REQUEST

After you create the VPC peering connection, you must accept the request in the Customer AWS Account.

Prerequisites

- Initiate the VPC peer request.

Procedure

1. Log in to the AWS Web Console.
2. Navigate to **VPC Service**.
3. Go to **Peering Connections**.
4. Click on **Pending peering connection**.
5. Confirm the AWS Account and VPC ID that the request originated from. This should be from the OSD AWS Account and OpenShift Dedicated Cluster VPC.

6. Click **Accept Request**.

3.4. CONFIGURING THE ROUTING TABLES

After you accept the VPC peering request, both VPCs must configure their routes to communicate across the peering connection.

Prerequisites

- Initiate and accept the VPC peer request.

Procedure

1. Log in to the AWS Web Console for the OSD AWS Account.
2. Navigate to the **VPC Service**, then **Route Tables**.
3. Select the Route Table for the OpenShift Dedicated Cluster VPC.

NOTE

On some clusters, there may be more than one route table for a particular VPC. Select the private one that has a number of explicitly associated subnets.

4. Select the **Routes** tab, then **Edit**.
5. Enter the Customer VPC CIDR block in the **Destination** text box.
6. Enter the Peering Connection ID in the **Target** text box.
7. Click **Save**.
8. You must complete the same process with the other VPC's CIDR block:
    a. Log into the Customer AWS Web Console → **VPC Service** → **Route Tables**.
    b. Select the Route Table for your VPC.
    c. Select the **Routes** tab, then **Edit**.
    d. Enter the OpenShift Dedicated Cluster VPC CIDR block in the **Destination** text box.
    e. Enter the Peering Connection ID in the **Target** text box.
    f. Click **Save**.

The VPC peering connection is now complete. Follow the verification procedure to ensure connectivity across the peering connection is working.

3.5. VERIFYING AND TROUBLESHOOTING VPC PEERING

After you set up a VPC peering connection, it is best to confirm it has been configured and is working correctly.

Prerequisites

- Initiate and accept the VPC peer request.
- Configure the routing tables.

Procedure

- In the AWS console, look at the route table for the cluster VPC that is peered. Ensure that the steps for configuring the routing tables were followed and that there is a route table entry pointing the VPC CIDR range destination to the peering connection target.
  If the routes look correct on both the OpenShift Dedicated Cluster VPC route table and Customer VPC route table, then the connection should be tested using the **netcat** method below. If the test calls are successful, then VPC peering is working correctly.
- To test network connectivity to an endpoint device, **nc** (or **netcat**) is a helpful troubleshooting tool. It is included in the default image and provides quick and clear output if a connection can be established:

  a. Create a temporary Pod using the **busybox** image, which cleans up after itself:

  ```
  $ oc run netcat-test \
      --image=busybox -i -t \
      --restart=Never --rm \
      -- /bin/sh
  ```

  b. Check the connection using **nc**.

  - Example successful connection results:

  ```
  / nc -zvv 192.168.1.1 8080
  10.181.3.180 (10.181.3.180:8080) open
  sent 0, rcvd 0
  ```

  - Example failed connection results:

  ```
  / nc -zvv 192.168.1.2 8080
  nc: 10.181.3.180 (10.181.3.180:8081): Connection refused
  sent 0, rcvd 0
  ```

  c. Exit the container, which automatically deletes the Pod:

  ```
  / exit
  ```

CHAPTER 4. CONFIGURING AWS VPN

This sample process configures an Amazon Web Services (AWS) OpenShift Dedicated cluster to use a customer's on-site hardware VPN device.

NOTE

AWS VPN does not currently provide a managed option to apply NAT to VPN traffic. See the AWS Knowledge Center for more details.

NOTE

Routing all traffic, for example **0.0.0.0/0**, through a private connection is not supported. This requires deleting the internet gateway, which disables SRE management traffic.

For more information about connecting an AWS VPC to remote networks using a hardware VPN device, see the Amazon VPC VPN Connections documentation.

4.1. CREATING A VPN CONNECTION

You can configure an Amazon Web Services (AWS) OpenShift Dedicated cluster to use a customer's on-site hardware VPN device using the following procedures.

Prerequisites

- Hardware VPN gateway device model and software version, for example Cisco ASA running version 8.3. See the Amazon VPC Network Administrator Guide to confirm whether your gateway device is supported by AWS.
- Public, static IP address for the VPN gateway device.
- BGP or static routing: if BGP, the ASN is required. If static routing, you must configure at least one static route.
- Optional: IP and Port/Protocol of a reachable service to test the VPN connection.

4.1.1. Configuring the VPN connection

Procedure

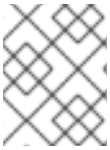
1. Log in to the OSD AWS Account Dashboard, and navigate to the VPC Dashboard.
2. Click on **Your VPCs** and identify the name and VPC ID for the VPC containing the OpenShift Dedicated cluster.
3. From the VPC Dashboard, click **Customer Gateway**.
4. Click **Create Customer Gateway** and give it a meaningful name.
5. Select the routing method: **Dynamic** or **Static**.
6. If Dynamic, enter the BGP ASN in the field that appears.
7. Paste in the VPN gateway endpoint IP address.

8. Click **Create**.
9. If you do not already have a Virtual Private Gateway attached to the intended VPC:
    a. From the VPC Dashboard, click on **Virtual Private Gateway**.
    b. Click **Create Virtual Private Gateway**, give it a meaningful name, and click **Create**.
    c. Leave the default Amazon default ASN.
    d. Select the newly created gateway, click **Attach to VPC**, and attach it to the cluster VPC you identified earlier.

4.1.2. Establishing the VPN Connection

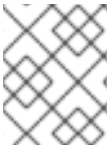
Procedure

1. From the VPC dashboard, click on **Site-to-Site VPN Connections**.
2. Click **Create VPN Connection**.
    a. Give it a meaningful name tag.
    b. Select the virtual private gateway created previously.
    c. For Customer Gateway, select **Existing**.
    d. Select the customer gateway device by name.
    e. If the VPN will use BGP, select **Dynamic**, otherwise select **Static**. Enter Static IP CIDRs. If there are multiple CIDRs, add each CIDR as **Another Rule**.
    f. Click **Create**.
    g. Wait for VPN status to change to **Available**, approximately 5 to 10 minutes.
3. Select the VPN you just created and click **Download Configuration**.
    a. From the dropdown list, select the vendor, platform, and version of the customer gateway device, then click **Download**.
    b. The **Generic** vendor configuration is also available for retrieving information in a plain text format.



NOTE

After the VPN connection has been established, be sure to set up Route Propagation or the VPN may not function as expected.



NOTE

Note the VPC subnet information, which you must add to your configuration as the remote network.

4.1.3. Enabling VPN route propagation

After you have set up the VPN connection, you must ensure that route propagation is enabled so that the necessary routes are added to the VPC's route table.

Procedure

1. From the VPC Dashboard, click on **Route Tables**.
2. Select the private Route table associated with the VPC that contains your OpenShift Dedicated cluster.

NOTE

On some clusters, there may be more than one route table for a particular VPC. Select the private one that has a number of explicitly associated subnets.

3. Click on the **Route Propagation** tab.
4. In the table that appears, you should see the virtual private gateway you created previously. Check the value in the **Propagate column**.
    a. If Propagate is set to **No**, click **Edit route propagation**, check the Propagate checkbox next to the virtual private gateway's name and click **Save**.

After you configure your VPN tunnel and AWS detects it as **Up**, your static or BGP routes are automatically added to the route table.

4.2. VERIFYING THE VPN CONNECTION

After you have set up your side of the VPN tunnel, you can verify that the tunnel is up in the AWS console and that connectivity across the tunnel is working.

Prerequisites

- Created a VPN connection.

Procedure

1. **Verify the tunnel is up in AWS.**
    a. From the VPC Dashboard, click on **VPN Connections**.
    b. Select the VPN connection you created previously and click the **Tunnel Details** tab.
    c. You should be able to see that at least one of the VPN tunnels is **Up**.
2. **Verify the connection.**
    To test network connectivity to an endpoint device, **nc** (or **netcat**) is a helpful troubleshooting tool. It is included in the default image and provides quick and clear output if a connection can be established:
    a. Create a temporary Pod using the **busybox** image, which cleans up after itself:

```
$ oc run netcat-test \
    --image=busybox -i -t \
    --restart=Never --rm \
```

```
-- /bin/sh
```

b. Check the connection using **nc**.

- Example successful connection results:

```
/ nc -zvv 192.168.1.1 8080
10.181.3.180 (10.181.3.180:8080) open
sent 0, rcvd 0
```

- Example failed connection results:

```
/ nc -zvv 192.168.1.2 8080
nc: 10.181.3.180 (10.181.3.180:8081): Connection refused
sent 0, rcvd 0
```

c. Exit the container, which automatically deletes the Pod:

```
/ exit
```

4.3. TROUBLESHOOTING THE VPN CONNECTION

Tunnel does not connect

If the tunnel connection is still **Down**, there are several things you can verify:

- The AWS tunnel will not initiate a VPN connection. The connection attempt must be initiated from the Customer Gateway.
- Ensure that your source traffic is coming from the same IP as the configured customer gateway. AWS will silently drop all traffic to the gateway whose source IP address does not match.
- Ensure that your configuration matches values supported by AWS. This includes IKE versions, DH groups, IKE lifetime, and more.
- Recheck the route table for the VPC. Ensure that propagation is enabled and that there are entries in the route table that have the virtual private gateway you created earlier as a target.
- Confirm that you do not have any firewall rules that could be causing an interruption.
- Check if you are using a policy-based VPN as this can cause complications depending on how it is configured.
- Further troubleshooting steps can be found at the AWS Knowledge Center.

Tunnel does not stay connected

If the tunnel connection has trouble staying **Up** consistently, know that all AWS tunnel connections must be initiated from your gateway. AWS tunnels do not initiate tunneling .

Red Hat recommends setting up an SLA Monitor (Cisco ASA) or some device on your side of the tunnel that constantly sends "interesting" traffic, for example **ping**, **nc**, or **telnet**, at any IP address configured within the VPC CIDR range. It does not matter whether the connection is successful, just that the traffic is being directed at the tunnel.

Secondary tunnel in Down state

When a VPN tunnel is created, AWS creates an additional failover tunnel. Depending upon the gateway device, sometimes the secondary tunnel will be seen as in the **Down** state.

The AWS Notification is as follows:

> You have new non-redundant VPN connections
>
> One or more of your vpn connections are not using both tunnels. This mode of operation is not highly available and we strongly recommend you configure your second tunnel. View your non-redundant VPN connections.

CHAPTER 5. CONFIGURING AWS DIRECT CONNECT

This process describes accepting an AWS Direct Connect virtual interface with OpenShift Dedicated. For more information about AWS Direct Connect types and configuration, see the AWS Direct Connect components documentation.

5.1. AWS DIRECT CONNECT METHODS

A Direct Connect connection requires a hosted Virtual Interface (VIF) connected to a Direct Connect Gateway (DXGateway), which is in turn associated to a Virtual Gateway (VGW) or a Transit Gateway in order to access a remote VPC in the same or another account.

If you do not have an existing DXGateway, the typical process involves creating the hosted VIF, with the DXGateway and VGW being created in the OSD AWS Account.

If you have an existing DXGateway connected to one or more existing VGWs, the process involves the OSD AWS Account sending an Association Proposal to the DXGateway owner. The DXGateway owner must ensure that the proposed CIDR will not conflict with any other VGWs they have associated.

See the following AWS documentation for more details:

- Virtual Interfaces
- Direct Connect Gateways
- Associating a VGW across accounts

IMPORTANT

When connecting to an existing DXGateway, you are responsible for the costs.

There are two configuration options available:

Method 1	Create the hosted VIF and then the DXGateway and VGW.
Method 2	Request a connection via an existing Direct Connect Gateway that you own.

5.2. CREATING THE HOSTED VIRTUAL INTERFACE

Prerequisites

- Gather OSD AWS Account ID.

5.2.1. Determining the type of Direct Connect connection

View the Direct Connect Virtual Interface details to determine the type of connection.

Procedure

1. Log in to the OSD AWS Account Dashboard and select the correct region.
2. Select **Direct Connect** from the **Services** menu.

3. There will be one or more Virtual Interfaces waiting to be accepted, select one of them to view the **Summary**.
4. View the Virtual Interface type: private or public.
5. Record the **Amazon side ASN** value.

If the Direct Connect Virtual Interface type is Private, a Virtual Private Gateway is created. If the Direct Connect Virtual Interface is Public, a Direct Connect Gateway is created.

5.2.2. Creating a Private Direct Connect

A Private Direct Connect is created if the Direct Connect Virtual Interface type is Private.

Procedure

1. Log in to the OSD AWS Account Dashboard and select the correct region.
2. From the AWS region, select **VPC** from the **Services** menu.
3. Select **Virtual Private Gateways** from **VPN Connections**.
4. Click **Create Virtual Private Gateway**.
5. Give the Virtual Private Gateway a suitable name.
6. Select **Custom ASN** and enter the **Amazon side ASN** value gathered previously.
7. Create the Virtual Private Gateway.
8. Click the newly created Virtual Private Gateway and choose **Attach to VPC** from the **Actions** tab.
9. Select the **OSD Cluster VPC** from the list, and attach the Virtual Private Gateway to the VPC.
10. From the **Services** menu, click **Direct Connect**. Choose one of the Direct Connect Virtual Interfaces from the list.
11. Acknowledge the **I understand that Direct Connect port charges apply once I click Accept Connection** message, then choose **Accept Connection**.
12. Choose to **Accept** the Virtual Private Gateway Connection and select the Virtual Private Gateway that was created in the previous steps.
13. Select **Accept** to accept the connection.
14. Repeat the previous steps if there is more than one Virtual Interface.

5.2.3. Creating a Public Direct Connect

A Public Direct Connect is created if the Direct Connect Virtual Interface type is Public.

Procedure

1. Log in to the OSD AWS Account Dashboard and select the correct region.

2. From the OSD AWS Account region, select **Direct Connect** from the **Services** menu.
3. Select **Direct Connect Gateways** and **Create Direct Connect Gateway**.
4. Give the Direct Connect Gateway a suitable name.
5. In the **Amazon side ASN**, enter the Amazon side ASN value gathered previously.
6. Create the Direct Connect Gateway.
7. Select **Direct Connect** from the **Services** menu.
8. Select one of the Direct Connect Virtual Interfaces from the list.
9. Acknowledge the **I understand that Direct Connect port charges apply once I click Accept Connection** message, then choose **Accept Connection**.
10. Choose to **Accept** the Direct Connect Gateway Connection and select the Direct Connect Gateway that was created in the previous steps.
11. Click **Accept** to accept the connection.
12. Repeat the previous steps if there is more than one Virtual Interface.

5.2.4. Verifying the Virtual Interfaces

After the Direct Connect Virtual Interfaces have been accepted, wait a short period and view the status of the Interfaces.

Procedure

1. Log in to the OSD AWS Account Dashboard and select the correct region.
2. From the OSD AWS Account region, select **Direct Connect** from the **Services** menu.
3. Select one of the Direct Connect Virtual Interfaces from the list.
4. Check the Interface State has become **Available**
5. Check the Interface BGP Status has become **Up**.
6. Repeat this verification for any remaining Direct Connect Interfaces.

After the Direct Connect Virtual Interfaces are available, you can log in to the OSD AWS Account Dashboard and download the Direct Connect configuration file for configuration on your side.

5.3. CONNECTING TO AN EXISTING DIRECT CONNECT GATEWAY

Prerequisites

- Confirm the CIDR range of the OSD VPC will not conflict with any other VGWs you have associated.
- Gather the following information:
  - The Direct Connect Gateway ID.

- The AWS Account ID associated with the virtual interface.
- The BGP ASN assigned for the DXGateway. Optional: the Amazon default ASN may also be used.

Procedure

1. Log in to the OSD AWS Account Dashboard and select the correct region.
2. From the OSD AWS Account region, select **VPC** from the **Services** menu.
3. From **VPN Connections**, select **Virtual Private Gateways**.
4. Select **Create Virtual Private Gateway**.
5. Give the Virtual Private Gateway a suitable name.
6. Click **Custom ASN** and enter the **Amazon side ASN** value gathered previously or use the Amazon Provided ASN.
7. Create the Virtual Private Gateway.
8. In the **Navigation** pane of the OSD AWS Account Dashboard, choose **Virtual private gateways** and select the virtual private gateway. Choose **View details**.
9. Choose **Direct Connect gateway associations** and click **Associate Direct Connect gateway**.
10. Under **Association account type**, for Account owner, choose **Another account**.
11. For **Direct Connect gateway owner**, enter the ID of the AWS account that owns the Direct Connect gateway.
12. Under **Association settings**, for Direct Connect gateway ID, enter the ID of the Direct Connect gateway.
13. Under **Association settings**, for Virtual interface owner, enter the ID of the AWS account that owns the virtual interface for the association.
14. Optional: Add prefixes to Allowed prefixes, separating them using commas.
15. Choose **Associate Direct Connect gateway**.
16. After the Association Proposal has been sent, it will be waiting for your acceptance. The final steps you must perform are available in the AWS Documentation.

5.4. TROUBLESHOOTING DIRECT CONNECT

Further troubleshooting can be found in the Troubleshooting AWS Direct Connect documentation.

CHAPTER 6. CONFIGURING A PRIVATE CLUSTER

An OpenShift Dedicated cluster can be made private so that internal applications can be hosted inside a corporate network. In addition, private clusters can be configured to have only internal API endpoints for increased security.

OpenShift Dedicated administrators can choose between public and private cluster configuration from within the **OpenShift Cluster Manager** (OCM). Privacy settings can be configured during cluster creation or after a cluster is established.

6.1. ENABLING PRIVATE CLUSTER ON A NEW CLUSTER

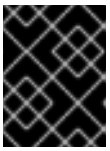
You can enable private cluster settings when creating a new cluster:

Prerequisites

- AWS VPC Peering, VPN, DirectConnect, or TransitGateway has been configured to allow private access.

Procedure

1. In the OpenShift Cluster Manager, click **Create cluster** and select **OpenShift Dedicated**.
2. Configure your cluster details, then select **Advanced** in the Networking section.
3. Determine your CIDR requirements for your network and input the required fields.



IMPORTANT

CIDR configurations cannot be changed later. Confirm your selections with your network administrator before proceeding.

4. Under **Cluster Privacy**, select **Private**.

6.2. ENABLING PRIVATE CLUSTER ON AN EXISTING CLUSTER

You can enable private clusters after a cluster has been created:

Prerequisites

- AWS VPC Peering, VPN, DirectConnect, or TransitGateway has been configured to allow private access.

Procedure

1. Access your cluster in the OpenShift Cluster Manager.
2. Navigate to the **Networking** tab.
3. Select **Make API private** under **Master API endpoint** and click **Change settings**.

NOTE

Transitioning your cluster between private and public can take several minutes to complete.

6.3. ENABLING PUBLIC CLUSTER ON A PRIVATE CLUSTER

You can set a private cluster to public facing:

Procedure

1. Access your cluster in the OpenShift Cluster Manager.
2. Navigate to the **Networking** tab.
3. Deselect **Make API private** under **Master API endpoint** and click **Change settings**.

NOTE

Transitioning your cluster between private and public can take several minutes to complete.

NOTE

Red Hat Service Reliability Engineers (SREs) can access a public or private cluster through the **cloud-ingress-operator** and existing ElasticSearch Load Balancer or Amazon S3 framework. SREs can access clusters through a secure endpoint to perform maintenance and service tasks.