Red Hat OpenShift Service on AWS 4

Cluster administration

Configuring Red Hat OpenShift Service on AWS clusters

Last Updated: 2023-05-25

Red Hat OpenShift Service on AWS 4 Cluster administration

Configuring Red Hat OpenShift Service on AWS clusters

Legal Notice

Abstract

This document provides information about configuring Red Hat OpenShift Service on AWS (ROSA) clusters.

Table of Contents

CHAPTER 1. CONFIGURING PRIVATE CONNECTIONS

1.1. CONFIGURING PRIVATE CONNECTIONS

Private cluster access can be implemented to suit the needs of your Red Hat OpenShift Service on AWS (ROSA) environment.

Procedure

1. Access your ROSA AWS account and 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.
2. Configure a private cluster on ROSA.

1.2. CONFIGURING AWS VPC PEERING

This sample process configures an Amazon Web Services (AWS) VPC containing an Red Hat OpenShift Service on AWS 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.

1.2.1. VPC peering terms

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

Red Hat OpenShift Service on AWS AWS Account	The AWS account that contains the Red Hat OpenShift Service on AWS cluster.
Red Hat OpenShift Service on AWS Cluster VPC	The VPC that contains the Red Hat OpenShift Service on AWS cluster.
Customer AWS Account	Your non-Red Hat OpenShift Service on AWS 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.

1.2.2. Initiating the VPC peer request

You can send a VPC peering connection request from the Red Hat OpenShift Service on AWS 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 Red Hat OpenShift Service on AWS 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 continue with the procedure.

Procedure

1. Log in to the Web Console for the Red Hat OpenShift Service on AWS 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 Red Hat OpenShift Service on AWS 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.

1.2.3. 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 Red Hat OpenShift Service on AWS AWS Account and Red Hat OpenShift Service on AWS Cluster VPC.
6. Click **Accept Request**.

1.2.4. 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 Red Hat OpenShift Service on AWS AWS Account.
2. Navigate to the **VPC Service**, then **Route Tables**.
3. Select the Route Table for the Red Hat OpenShift Service on AWS 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 Red Hat OpenShift Service on AWS 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.

1.2.5. 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 Red Hat OpenShift Service on AWS 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:

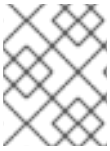
```
/ nc -zvv 192.168.1.2 8080
nc: 10.181.3.180 (10.181.3.180:8081): Connection refused
sent 0, rcvd 0
```

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

```
/ exit
```

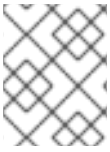
1.3. CONFIGURING AWS VPN

This sample process configures an Amazon Web Services (AWS) Red Hat OpenShift Service on AWS 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.

1.3.1. Creating a VPN connection

You can configure an Amazon Web Services (AWS) Red Hat OpenShift Service on AWS 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.

1.3.1.1. Configuring the VPN connection

Procedure

1. Log in to the Red Hat OpenShift Service on AWS 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 Red Hat OpenShift Service on AWS 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.

1.3.1.2. 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.

1.3.1.3. 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 Red Hat OpenShift Service on AWS 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.

1.3.2. 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
   ```

1.3.3. 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.

1.4. CONFIGURING AWS DIRECT CONNECT

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

1.4.1. 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 Red Hat OpenShift Service on AWS AWS Account.

If you have an existing DXGateway connected to one or more existing VGWs, the process involves the Red Hat OpenShift Service on AWS 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.

1.4.2. Creating the hosted Virtual Interface

Prerequisites

- Gather Red Hat OpenShift Service on AWS AWS Account ID.

1.4.2.1. 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 Red Hat OpenShift Service on AWS 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.

1.4.2.2. 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 Red Hat OpenShift Service on AWS 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 **Red Hat OpenShift Service on AWS 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.

1.4.2.3. 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 Red Hat OpenShift Service on AWS AWS Account Dashboard and select the correct region.
2. From the Red Hat OpenShift Service on AWS 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.

1.4.2.4. 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 Red Hat OpenShift Service on AWS AWS Account Dashboard and select the correct region.
2. From the Red Hat OpenShift Service on AWS 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 Red Hat OpenShift Service on AWS AWS Account Dashboard and download the Direct Connect configuration file for configuration on your side.

1.4.3. Connecting to an existing Direct Connect Gateway

Prerequisites

- Confirm the CIDR range of the Red Hat OpenShift Service on AWS 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 Red Hat OpenShift Service on AWS AWS Account Dashboard and select the correct region.
2. From the Red Hat OpenShift Service on AWS 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 Red Hat OpenShift Service on AWS 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.

1.4.4. Troubleshooting Direct Connect

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

CHAPTER 2. NODES

2.1. ABOUT MACHINE POOLS

Red Hat OpenShift Service on AWS uses machine pools as an elastic, dynamic provisioning method on top of your cloud infrastructure.

The primary resources are machines, compute machine sets, and machine pools.

IMPORTANT

As of the Red Hat OpenShift Service on AWS versions 4.8.35, 4.9.26, 4.10.6, the Red Hat OpenShift Service on AWS default per-pod pid limit is **4096**. If you want to enable this PID limit, you must upgrade your Red Hat OpenShift Service on AWS clusters to these versions or later. Red Hat OpenShift Service on AWS clusters with prior versions use a default PID limit of **1024**.

You cannot configure the per-pod PID limit on any Red Hat OpenShift Service on AWS cluster.

2.1.1. Machines

A machine is a fundamental unit that describes the host for a worker node.

2.1.2. Machine sets

MachineSet resources are groups of compute machines. If you need more machines or must scale them down, change the number of replicas in the machine pool to which the compute machine sets belong.

Machine sets are not directly modifiable in ROSA.

2.1.3. Machine pools

Machine pools are a higher level construct to compute machine sets.

A machine pool creates compute machine sets that are all clones of the same configuration across availability zones. Machine pools perform all of the host node provisioning management actions on a worker node. If you need more machines or must scale them down, change the number of replicas in the machine pool to meet your compute needs. You can manually configure scaling or set autoscaling.

By default, a cluster is created with one machine pool. You can add additional machine pools to an existing cluster, modify the default machine pool, and delete machine pools.

Multiple machine pools can exist on a single cluster, and they can each have different types or different size nodes.

2.1.4. Machine pools in multiple zone clusters

When you create a machine pool in a multiple availability zone (Multi-AZ) cluster, that one machine pool has 3 zones. The machine pool, in turn, creates a total of 3 compute machine sets - one for each zone in the cluster. Each of those compute machine sets manages one or more machines in its respective availability zone.

If you create a new Multi-AZ cluster, the machine pools are replicated to those zones automatically. If

you add a machine pool to an existing Multi-AZ, the new pool is automatically created in those zones. Similarly, deleting a machine pool will delete it from all zones. Due to this multiplicative effect, using machine pools in Multi-AZ cluster can consume more of your project's quota for a specific region when creating machine pools.

2.1.5. Additional resources

- Managing worker nodes
- About autoscaling

2.2. MANAGING COMPUTE NODES

This document describes how to manage compute (also known as worker) nodes with Red Hat OpenShift Service on AWS (ROSA).

The majority of changes for compute nodes are configured on machine pools. A machine pool is a group of compute nodes in a cluster that have the same configuration, providing ease of management.

You can edit machine pool configuration options such as scaling, adding node labels, and adding taints.

2.2.1. Creating a machine pool

A default machine pool is created when you install a Red Hat OpenShift Service on AWS (ROSA) cluster. After installation, you can create additional machine pools for your cluster by using OpenShift Cluster Manager or the ROSA CLI (**rosa**).

2.2.1.1. Creating a machine pool using OpenShift Cluster Manager

You can create additional machine pools for your Red Hat OpenShift Service on AWS (ROSA) cluster by using OpenShift Cluster Manager.

Prerequisites

- You created a ROSA cluster.

Procedure

1. Navigate to OpenShift Cluster Manager Hybrid Cloud Console and select your cluster.
2. Under the **Machine pools** tab, click **Add machine pool**.
3. Add a **Machine pool name**.
4. Select a **Worker node instance type** from the drop-down menu. The instance type defines the vCPU and memory allocation for each compute node in the machine pool.

NOTE

You cannot change the instance type for a machine pool after the pool is created.

5. Optional: Configure autoscaling for the machine pool:

a. Select **Enable autoscaling** to automatically scale the number of machines in your machine pool to meet the deployment needs.

b. Set the minimum and maximum node count limits for autoscaling. The cluster autoscaler does not reduce or increase the machine pool node count beyond the limits that you specify.

- If you deployed your cluster using a single availability zone, set the **Minimum and maximum node count**. This defines the minimum and maximum compute node limits in the availability zone.
- If you deployed your cluster using multiple availability zones, set the **Minimum nodes per zone** and **Maximum nodes per zone**. This defines the minimum and maximum compute node limits per zone.

NOTE

Alternatively, you can set your autoscaling preferences for the machine pool after the machine pool is created.

6. If you did not enable autoscaling, select a compute node count:

- If you deployed your cluster using a single availability zone, select a **Worker node count** from the drop-down menu. This defines the number of compute nodes to provision to the machine pool for the zone.
- If you deployed your cluster using multiple availability zones, select a **Worker node count (per zone)** from the drop-down menu. This defines the number of compute nodes to provision to the machine pool per zone.

7. Optional: Add node labels and taints for your machine pool:

a. Expand the **Edit node labels and taints** menu.

b. Under **Node labels**, add **Key** and **Value** entries for your node labels.

c. Under **Taints**, add **Key** and **Value** entries for your taints.

d. For each taint, select an **Effect** from the drop-down menu. Available options include **NoSchedule**, **PreferNoSchedule**, and **NoExecute**.

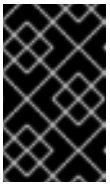
NOTE

Alternatively, you can add the node labels and taints after you create the machine pool.

8. Optional: Use Amazon EC2 Spot Instances if you want to configure your machine pool to deploy machines as non-guaranteed AWS Spot Instances:

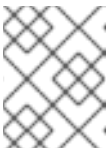
a. Select **Use Amazon EC2 Spot Instances**.

b. Leave **Use On-Demand instance price** selected to use the on-demand instance price. Alternatively, select **Set maximum price** to define a maximum hourly price for a Spot Instance.
For more information about Amazon EC2 Spot Instances, see the AWS documentation.



IMPORTANT

Your Amazon EC2 Spot Instances might be interrupted at any time. Use Amazon EC2 Spot Instances only for workloads that can tolerate interruptions.



NOTE

If you select **Use Amazon EC2 Spot Instances** for a machine pool, you cannot disable the option after the machine pool is created.

9. Click **Add machine pool** to create the machine pool.

Verification

- Verify that the machine pool is visible on the **Machine pools** page and the configuration is as expected.

2.2.1.2. Creating a machine pool using the ROSA CLI

You can create additional machine pools for your Red Hat OpenShift Service on AWS (ROSA) cluster by using the ROSA CLI (**rosa**).

Prerequisites

- You installed and configured the latest AWS (**aws**), ROSA (**rosa**), and OpenShift (**oc**) CLIs on your workstation.
- You logged in to your Red Hat account by using the **rosa** CLI.
- You created a ROSA cluster.

Procedure

- To add a machine pool that does not use autoscaling, create the machine pool and define the instance type, compute (also known as worker) node count, and node labels:

```
$ rosa create machinepool --cluster=<cluster-name> \
                          --name=<machine_pool_id> \ 1
                          --replicas=<replica_count> \ 2
                          --instance-type=<instance_type> \ 3
                          --labels=<key>=<value>,<key>=<value> \ 4
                          --taints=<key>=<value>:<effect>,<key>=<value>:<effect> \ 5
                          --use-spot-instances \ 6
                          --spot-max-price=0.5 7
```

1. Specifies the name of the machine pool. Replace **<machine_pool_id>** with the name of your machine pool.
2. Specifies the number of compute nodes to provision. If you deployed ROSA using a single availability zone, this defines the number of compute nodes to provision to the machine pool for the zone. If you deployed your cluster using multiple availability zones, this defines the number of compute nodes to provision in total across all zones and the count must be a

multiple of 3. The **--replicas** argument is required when autoscaling is not configured.

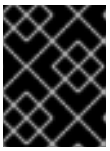
3. Optional: Sets the instance type for the compute nodes in your machine pool. The instance type defines the vCPU and memory allocation for each compute node in the pool. Replace **<instance_type>** with an instance type. The default is **m5.xlarge**. You cannot change the instance type for a machine pool after the pool is created.

4. Optional: Defines the labels for the machine pool. Replace **<key>=<value>,<key>=<value>** with a comma-delimited list of key-value pairs, for example **--labels=key1=value1,key2=value2**.

5. Optional: Defines the taints for the machine pool. Replace **<key>=<value>:<effect>, <key>=<value>:<effect>** with a key, value, and effect for each taint, for example **--taints=key1=value1:NoSchedule,key2=value2:NoExecute**. Available effects include **NoSchedule**, **PreferNoSchedule**, and **NoExecute**.

6. Optional: Configures your machine pool to deploy machines as non-guaranteed AWS Spot Instances. For information, see Amazon EC2 Spot Instances in the AWS documentation. If you select **Use Amazon EC2 Spot Instances** for a machine pool, you cannot disable the option after the machine pool is created.

7. Optional: If you have opted to use Spot Instances, you can specify this argument to define a maximum hourly price for a Spot Instance. If this argument is not specified, the on-demand price is used.



IMPORTANT

Your Amazon EC2 Spot Instances might be interrupted at any time. Use Amazon EC2 Spot Instances only for workloads that can tolerate interruptions.

The following example creates a machine pool called **mymachinepool** that uses the **m5.xlarge** instance type and has 2 compute node replicas. The example also adds 2 workload-specific labels:

```
$ rosa create machinepool --cluster=mycluster --name=mymachinepool --replicas=2 --
instance-type=m5.xlarge --labels=app=db,tier=backend
```

Example output

```
I: Machine pool 'mymachinepool' created successfully on cluster 'mycluster'
I: To view all machine pools, run 'rosa list machinepools -c mycluster'
```

- To add a machine pool that uses autoscaling, create the machine pool and define the autoscaling configuration, instance type and node labels:

```
$ rosa create machinepool --cluster=<cluster-name> \
                --name=<machine_pool_id> \ 1
                --enable-autoscaling \ 2
                --min-replicas=<minimum_replica_count> \ 3
                --max-replicas=<maximum_replica_count> \ 4
                --instance-type=<instance_type> \ 5
                --labels=<key>=<value>,<key>=<value> \ 6
```

```
--taints=<key>=<value>:<effect>,<key>=<value>:<effect> \ 7
--use-spot-instances \ 8
--spot-max-price=0.5 9
```

1. Specifies the name of the machine pool. Replace **<machine_pool_id>** with the name of your machine pool.
2. Enables autoscaling in the machine pool to meet the deployment needs.
3. 4. Defines the minimum and maximum compute node limits. The cluster autoscaler does not reduce or increase the machine pool node count beyond the limits that you specify. If you deployed ROSA using a single availability zone, the **--min-replicas** and **--max-replicas** arguments define the autoscaling limits in the machine pool for the zone. If you deployed your cluster using multiple availability zones, the arguments define the autoscaling limits in total across all zones and the counts must be multiples of 3.
5. Optional: Sets the instance type for the compute nodes in your machine pool. The instance type defines the vCPU and memory allocation for each compute node in the pool. Replace **<instance_type>** with an instance type. The default is **m5.xlarge**. You cannot change the instance type for a machine pool after the pool is created.
6. Optional: Defines the labels for the machine pool. Replace **<key>=<value>,<key>=<value>** with a comma-delimited list of key-value pairs, for example **--labels=key1=value1,key2=value2**.
7. Optional: Defines the taints for the machine pool. Replace **<key>=<value>:<effect>, <key>=<value>:<effect>** with a key, value, and effect for each taint, for example **--taints=key1=value1:NoSchedule,key2=value2:NoExecute**. Available effects include **NoSchedule**, **PreferNoSchedule**, and **NoExecute**.
8. Optional: Configures your machine pool to deploy machines as non-guaranteed AWS Spot Instances. For information, see Amazon EC2 Spot Instances in the AWS documentation. If you select **Use Amazon EC2 Spot Instances** for a machine pool, you cannot disable the option after the machine pool is created.
9. Optional: If you have opted to use Spot Instances, you can specify this argument to define a maximum hourly price for a Spot Instance. If this argument is not specified, the on-demand price is used.

IMPORTANT

Your Amazon EC2 Spot Instances might be interrupted at any time. Use Amazon EC2 Spot Instances only for workloads that can tolerate interruptions.

The following example creates a machine pool called **mymachinepool** that uses the **m5.xlarge** instance type and has autoscaling enabled. The minimum compute node limit is 3 and the maximum is 6 overall. The example also adds 2 workload-specific labels:

```
$ rosa create machinepool --cluster=mycluster --name=mymachinepool --enable-autoscaling
--min-replicas=3 --max-replicas=6 --instance-type=m5.xlarge --labels=app=db,tier=backend
```

Example output

```
I: Machine pool 'mymachinepool' created successfully on cluster 'mycluster'
I: To view all machine pools, run 'rosa list machinepools -c mycluster'
```

Verification

1. List the available machine pools in your cluster:

```
$ rosa list machinepools --cluster=<cluster_name>
```

Example output

```
ID           AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS                TAINTS
AVAILABILITY ZONES                SPOT INSTANCES
Default      No           3         m5.xlarge                            us-east-1a, us-east-1b, us-
east-1c    N/A
mymachinepool  Yes        3-6       m5.xlarge      app=db, tier=backend  us-east-1a,
us-east-1b, us-east-1c    No
```

2. Verify that the machine pool is included in the output and the configuration is as expected.

Additional resources

- For a detailed list of the arguments that are available for the **rosa create machinepool** subcommand, see Managing objects with the rosa CLI.

2.2.2. Scaling compute nodes manually

If you have not enabled autoscaling for your machine pool, you can manually scale the number of compute (also known as worker) nodes in the pool to meet your deployment needs.

You must scale each machine pool separately.

Prerequisites

- You installed and configured the latest AWS (**aws**), ROSA (**rosa**), and OpenShift (**oc**) CLIs on your workstation.
- You logged in to your Red Hat account by using the **rosa** CLI.
- You created a Red Hat OpenShift Service on AWS (ROSA) cluster.
- You have an existing machine pool.

Procedure

1. List the machine pools in the cluster:

```
$ rosa list machinepools --cluster=<cluster_name>
```

Example output

```
ID       AUTOSCALING   REPLICAS    INSTANCE TYPE  LABELS    TAINTS
AVAILABILITY ZONES
```

```
default   No        2        m5.xlarge                   us-east-1a
mp1       No        2        m5.xlarge                   us-east-1a
```

2. Increase or decrease the number of compute node replicas in a machine pool:

```
$ rosa edit machinepool --cluster=<cluster_name> \
                        --replicas=<replica_count> \ 1
                        <machine_pool_id> 2
```

1. If you deployed Red Hat OpenShift Service on AWS (ROSA) using a single availability zone, the replica count defines the number of compute nodes to provision to the machine pool for the zone. If you deployed your cluster using multiple availability zones, the count defines the total number of compute nodes in the machine pool across all zones and must be a multiple of 3.

2. Replace **<machine_pool_id>** with the ID of your machine pool, as listed in the output of the preceding command.

Verification

1. List the available machine pools in your cluster:

```
$ rosa list machinepools --cluster=<cluster_name>
```

Example output

```
ID        AUTOSCALING   REPLICAS    INSTANCE TYPE  LABELS    TAINTS
AVAILABILITY ZONES
default   No            2           m5.xlarge                 us-east-1a
mp1       No            3           m5.xlarge                 us-east-1a
```

2. In the output of the preceding command, verify that the compute node replica count is as expected for your machine pool. In the example output, the compute node replica count for the **mp1** machine pool is scaled to 3.

2.2.3. Node labels

A label is a key-value pair applied to a **Node** object. You can use labels to organize sets of objects and control the scheduling of pods.

You can add labels during cluster creation or after. Labels can be modified or updated at any time.

Additional resources

- For more information about labels, see Kubernetes Labels and Selectors overview .

2.2.3.1. Adding node labels to a machine pool

Add or edit labels for compute (also known as worker) nodes at any time to manage the nodes in a manner that is relevant to you. For example, you can assign types of workloads to specific nodes.

Labels are assigned as key-value pairs. Each key must be unique to the object it is assigned to.

Prerequisites

- You installed and configured the latest AWS (**aws**), ROSA (**rosa**), and OpenShift (**oc**) CLIs on your workstation.
- You logged in to your Red Hat account by using the **rosa** CLI.
- You created a Red Hat OpenShift Service on AWS (ROSA) cluster.
- You have an existing machine pool.

Procedure

1. List the machine pools in the cluster:

```
$ rosa list machinepools --cluster=<cluster_name>
```

Example output

```
ID          AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS    TAINTS
AVAILABILITY ZONES    SPOT INSTANCES
Default     No           2         m5.xlarge                  us-east-1a          N/A
db-nodes-mp  No          2         m5.xlarge                  us-east-1a          No
```

2. Add or update the node labels for a machine pool:

   - To add or update node labels for a machine pool that does not use autoscaling, run the following command:

```
$ rosa edit machinepool --cluster=<cluster_name> \
                        --replicas=<replica_count> \ 1
                        --labels=<key>=<value>,<key>=<value> \ 2
                        <machine_pool_id>
```

   1 For machine pools that do not use autoscaling, you must provide a replica count when adding node labels. If you do not specify the **--replicas** argument, you are prompted for a replica count before the command completes. If you deployed Red Hat OpenShift Service on AWS (ROSA) using a single availability zone, the replica count defines the number of compute nodes to provision to the machine pool for the zone. If you deployed your cluster using multiple availability zones, the count defines the total number of compute nodes in the machine pool across all zones and must be a multiple of 3.

   2 Replace **<key>=<value>,<key>=<value>** with a comma-delimited list of key-value pairs, for example **--labels=key1=value1,key2=value2**. This list overwrites any modifications made to node labels on an ongoing basis.

   The following example adds labels to the **db-nodes-mp** machine pool:

```
$ rosa edit machinepool --cluster=mycluster --replicas=2 --labels=app=db,tier=backend
db-nodes-mp
```

Example output

```
I: Updated machine pool 'db-nodes-mp' on cluster 'mycluster'
```

- To add or update node labels for a machine pool that uses autoscaling, run the following command:

```
$ rosa edit machinepool --cluster=<cluster_name> \
                        --min-replicas=<minimum_replica_count> \ 1
                        --max-replicas=<maximum_replica_count> \ 2
                        --labels=<key>=<value>,<key>=<value> \ 3
                        <machine_pool_id>
```

1 2 For machine pools that use autoscaling, you must provide minimum and maximum compute node replica limits. If you do not specify the arguments, you are prompted for the values before the command completes. The cluster autoscaler does not reduce or increase the machine pool node count beyond the limits that you specify. If you deployed ROSA using a single availability zone, the **--min-replicas** and **--max-replicas** arguments define the autoscaling limits in the machine pool for the zone. If you deployed your cluster using multiple availability zones, the arguments define the autoscaling limits in total across all zones and the counts must be multiples of 3.

3 Replace **<key>=<value>,<key>=<value>** with a comma-delimited list of key-value pairs, for example **--labels=key1=value1,key2=value2**. This list overwrites any modifications made to node labels on an ongoing basis.

The following example adds labels to the **db-nodes-mp** machine pool:

```
$ rosa edit machinepool --cluster=mycluster --min-replicas=2 --max-replicas=3 --
labels=app=db,tier=backend db-nodes-mp
```

Example output

```
I: Updated machine pool 'db-nodes-mp' on cluster 'mycluster'
```

Verification

1. List the available machine pools in your cluster:

```
$ rosa list machinepools --cluster=<cluster_name>
```

Example output

```
ID          AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS                TAINTS
AVAILABILITY ZONES    SPOT INSTANCES
Default     No           2         m5.xlarge                            us-east-1a          N/A
db-nodes-mp  No          2         m5.xlarge      app=db, tier=backend  us-east-1a
No
```

2. Verify that the labels are included for your machine pool in the output.

2.2.4. Adding taints to a machine pool

You can add taints for compute (also known as worker) nodes in a machine pool to control which pods are scheduled to them. When you apply a taint to a machine pool, the scheduler cannot place a pod on the nodes in the pool unless the pod specification includes a toleration for the taint.

Prerequisites

- You installed and configured the latest AWS (**aws**), ROSA (**rosa**), and OpenShift (**oc**) CLIs on your workstation.
- You logged in to your Red Hat account by using the **rosa** CLI.
- You created a Red Hat OpenShift Service on AWS (ROSA) cluster.
- You have an existing machine pool.

Procedure

1. List the machine pools in the cluster:

```
$ rosa list machinepools --cluster=<cluster_name>
```

Example output

```
ID          AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS    TAINTS
AVAILABILITY ZONES    SPOT INSTANCES
Default     No           2         m5.xlarge                  us-east-1a          N/A
db-nodes-mp  No          2         m5.xlarge                  us-east-1a          No
```

2. Add or update the taints for a machine pool:

   - To add or update taints for a machine pool that does not use autoscaling, run the following command:

```
$ rosa edit machinepool --cluster=<cluster_name> \
                        --replicas=<replica_count> \ 1
                        --taints=<key>=<value>:<effect>,<key>=<value>:<effect> \ 2
                        <machine_pool_id>
```

1 For machine pools that do not use autoscaling, you must provide a replica count when adding taints. If you do not specify the **--replicas** argument, you are prompted for a replica count before the command completes. If you deployed Red Hat OpenShift Service on AWS (ROSA) using a single availability zone, the replica count defines the number of compute nodes to provision to the machine pool for the zone. If you deployed your cluster using multiple availability zones, the count defines the total number of compute nodes in the machine pool across all zones and must be a multiple of 3.

2 Replace **<key>=<value>:<effect>,<key>=<value>:<effect>** with a key, value, and effect for each taint, for example **--taints=key1=value1:NoSchedule,key2=value2:NoExecute**. Available effects include **NoSchedule**, **PreferNoSchedule**, and **NoExecute**.This list overwrites any modifications made to node taints on an ongoing basis.

The following example adds taints to the **db-nodes-mp** machine pool:

```
$ rosa edit machinepool --cluster=mycluster --replicas 2 --
taints=key1=value1:NoSchedule,key2=value2:NoExecute db-nodes-mp
```

Example output

```
I: Updated machine pool 'db-nodes-mp' on cluster 'mycluster'
```

- To add or update taints for a machine pool that uses autoscaling, run the following command:

```
$ rosa edit machinepool --cluster=<cluster_name> \
                        --min-replicas=<minimum_replica_count> \ 1
                        --max-replicas=<maximum_replica_count> \ 2
                        --taints=<key>=<value>:<effect>,<key>=<value>:<effect> \ 3
                        <machine_pool_id>
```

1 2 For machine pools that use autoscaling, you must provide minimum and maximum compute node replica limits. If you do not specify the arguments, you are prompted for the values before the command completes. The cluster autoscaler does not reduce or increase the machine pool node count beyond the limits that you specify. If you deployed ROSA using a single availability zone, the **--min-replicas** and **--max-replicas** arguments define the autoscaling limits in the machine pool for the zone. If you deployed your cluster using multiple availability zones, the arguments define the autoscaling limits in total across all zones and the counts must be multiples of 3.

3 Replace **<key>=<value>:<effect>,<key>=<value>:<effect>** with a key, value, and effect for each taint, for example **--taints=key1=value1:NoSchedule,key2=value2:NoExecute**. Available effects include **NoSchedule**, **PreferNoSchedule**, and **NoExecute**.This list overwrites any modifications made to node taints on an ongoing basis.

The following example adds taints to the **db-nodes-mp** machine pool:

```
$ rosa edit machinepool --cluster=mycluster --min-replicas=2 --max-replicas=3 --
taints=key1=value1:NoSchedule,key2=value2:NoExecute db-nodes-mp
```

Example output

```
I: Updated machine pool 'db-nodes-mp' on cluster 'mycluster'
```

Verification

1. List the available machine pools in your cluster:

```
$ rosa list machinepools --cluster=<cluster_name>
```

Example output

```
ID          AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS    TAINTS
AVAILABILITY ZONES    SPOT INSTANCES
Default     No           2         m5.xlarge                                  us-east-1a
```

```
N/A
db-nodes-mp  No          2          m5.xlarge              key1=value1:NoSchedule,
key2=value2:NoExecute    us-east-1a          No
```

2. Verify that the taints are included for your machine pool in the output.

2.2.5. Adding node tuning to a machine pool

You can add tunings for compute (also known as worker) nodes in a machine pool to control their configuration.

Prerequisites

- You installed and configured the latest AWS (**aws**), ROSA (**rosa**), and OpenShift (**oc**) CLIs on your workstation.
- You logged in to your Red Hat account by using the ROSA CLI.
- You created a Red Hat OpenShift Service on AWS (ROSA) cluster.
- You have an existing machine pool.
- You have an existing tuning configuration.

Procedure

1. List the machine pools in the cluster:

```
$ rosa list machinepools --cluster=<cluster_name>
```

Example output

```
ID          AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS    TAINTS
AVAILABILITY ZONES    SUBNET  VERSION  AUTOREPAIR  TUNING CONFIGS
MESSAGE
Default     No           2         m5.xlarge                    us-east-1a          N/A     4.12.14  Yes
db-nodes-mp  No          2         m5.xlarge                    us-east-1a          No      4.12.14
Yes
```

2. You can add tuning configurations to an existing or new machine pool.

   a. Add tunings when creating a machine pool:

```
$ rosa create machinepool -c <cluster-name> <machinepoolname> --tuning-configs
<tuning_config_name>
```

Example output

```
? Tuning configs: sample-tuning
I: Machine pool 'db-nodes-mp' created successfully on hosted cluster 'sample-cluster'
I: To view all machine pools, run 'rosa list machinepools -c sample-cluster'
```

   b. Add or update the tunings for a machine pool:

```
$ rosa edit machinepool -c <cluster-name> <machinepoolname> --tuning-configs
<tuningconfigname>
```

Example output

```
I: Updated machine pool 'db-nodes-mp' on cluster 'mycluster'
```

Verification

1. List the available machine pools in your cluster:

```
$ rosa list machinepools --cluster=<cluster_name>
```

Example output

```
ID          AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS  TAINTS  AVAILABILITY
ZONES  SUBNET  VERSION  AUTOREPAIR  TUNING CONFIGS MESSAGE
Default     No           2         m5.xlarge                      us-east-1a          N/A     4.12.14  Yes
db-nodes-mp  No          2         m5.xlarge                      us-east-1a          No      4.12.14  Yes
sample-tuning
```

2. Verify that the tuning config is included for your machine pool in the output.

2.2.6. Additional resources

- About machine pools
- About autoscaling
- Enabling autoscaling
- Disabling autoscaling
- ROSA Service Definition

2.3. CONFIGURING MACHINE POOLS IN LOCAL ZONES

This document describes how to configure Local Zones in machine pools with Red Hat OpenShift Service on AWS (ROSA).

2.3.1. Configuring machine pools in Local Zones

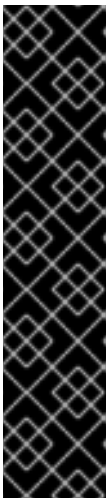
Use the following steps to configure machine pools in Local Zones.

IMPORTANT

AWS Local Zones are supported on Red Hat OpenShift Service on AWS 4.12. See the Red Hat Knowledgebase article for information on how to enable Local Zones.

Prerequisites

- ROSA is generally available in the parent region of choice. See the AWS generally available locations list to determine the Local Zone available to specific AWS regions.
- The ROSA cluster was initially built in an existing Amazon VPC (BYO-VPC).
- The maximum transmission unit (MTU) for the ROSA cluster is set at 1200.



IMPORTANT

Generally, the Maximum Transmission Unit (MTU) between an Amazon EC2 instance in a Local Zone and an Amazon EC2 instance in the Region is 1300. See How Local Zones work in the AWS documentation. The cluster network MTU must always be less than the EC2 MTU to account for the overhead. The specific overhead is determined by your network plugin, for example:

  - OVN-Kubernetes: **100 bytes**
  - OpenShift SDN: **50 bytes**

The network plugin could provide additional features that may also decrease the MTU. Check the documentation for additional information.

- The AWS account has Local Zones enabled.
- The AWS account has a Local Zone subnet for the same VPC as the cluster.
- The AWS account has a subnet that is associated with a routing table that has a route to a NAT gateway.
- The AWS account has the tag `kubernetes.io/cluster/<infra_id>: shared' on the associated subnet.

Procedure

1. Create a machine pool on the cluster by running the following ROSA CLI command.

```
$ rosa create machinepool -c <cluster-name> -i
```

2. Add the subnet and instance type for the machine pool in ROSA CLI. After several minutes, the cluster will provision the nodes.

```
I: Enabling interactive mode (1)
? Machine pool name: xx-lz-xx (2)
? Create multi-AZ machine pool: No (3)
? Select subnet for a single AZ machine pool (optional): Yes (4)
? Subnet ID: subnet-<a> (region-info) (5)
? Enable autoscaling (optional): No (6)
? Replicas: 2 (7)
I: Fetching instance types (8)
```

(1) Enables interactive mode.

(2) Names the machine pool. This is limited to alphanumeric and a maximum length of 30 characters.

3. Set this option to no.
4. Set this option to yes.
5. Selects a subnet ID from the list.
6. Select yes to enable autoscaling or no to disable autoscaling.
7. Selects the number of machines for the machine pool. This number can be anywhere from 1 - 180.
8. Selects an instance type from the list. Only instance types that are supported in the selected Local Zone will appear.

3. Provide the subnet ID to provision the machine pool in the Local Zone.

See the AWS Local Zones locations list on AWS for generally available and announced AWS Local Zone locations.

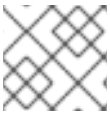
2.4. ABOUT AUTOSCALING NODES ON A CLUSTER

The autoscaler option can be configured to automatically scale the number of machines in a cluster.

The cluster autoscaler increases the size of the cluster when there are pods that failed to schedule on any of the current nodes due to insufficient resources or when another node is necessary to meet deployment needs. The cluster autoscaler does not increase the cluster resources beyond the limits that you specify.

Additionally, the cluster autoscaler decreases the size of the cluster when some nodes are consistently not needed for a significant period, such as when it has low resource use and all of its important pods can fit on other nodes.

When you enable autoscaling, you must also set a minimum and maximum number of worker nodes.



NOTE

Only cluster owners and organization admins can scale or delete a cluster.

2.4.1. Enabling autoscaling nodes on a cluster

You can enable autoscaling on worker nodes to increase or decrease the number of nodes available by editing the machine pool definition for an existing cluster.

Enabling autoscaling nodes in an existing cluster using Red Hat OpenShift Cluster Manager
Enable autoscaling for worker nodes in the machine pool definition from OpenShift Cluster Manager console.

Procedure

1. From OpenShift Cluster Manager Hybrid Cloud Console , navigate to the **Clusters** page and select the cluster that you want to enable autoscaling for.
2. On the selected cluster, select the **Machine pools** tab.

3. Click the Options menu  at the end of the machine pool that you want to enable autoscaling for and select **Scale**.

4. On the **Edit node count** dialog, select the **Enable autoscaling** checkbox.

5. Select **Apply** to save these changes and enable autoscaling for the cluster.

NOTE

Additionally, you can configure autoscaling on the default machine pool when you create the cluster using interactive mode.

Enabling autoscaling nodes in an existing cluster using the rosa CLI
Configure autoscaling to dynamically scale the number of worker nodes up or down based on load.

Successful autoscaling is dependent on having the correct AWS resource quotas in your AWS account. Verify resource quotas and request quota increases from the AWS console.

Procedure

1. To identify the machine pool IDs in a cluster, enter the following command:

```
$ rosa list machinepools --cluster=<cluster_name>
```

Example output

```
ID       AUTOSCALING   REPLICAS   INSTANCE TYPE  LABELS   TINTS   AVAILABILITY ZONES
default  No            2          m5.xlarge                      us-east-1a
mp1      No            2          m5.xlarge                      us-east-1a
```

2. Get the ID of the machine pools that you want to configure.

3. To enable autoscaling on a machine pool, enter the following command:

```
$ rosa edit machinepool --cluster=<cluster_name> <machinepool_ID> --enable-autoscaling --min-replicas=<number> --max-replicas=<number>
```

Example

Enable autoscaling on a machine pool with the ID **mp1** on a cluster named **mycluster**, with the number of replicas set to scale between 2 and 5 worker nodes:

```
$ rosa edit machinepool --cluster=mycluster mp1 --enable-autoscaling --min-replicas=2 --max-replicas=5
```

2.4.2. Disabling autoscaling nodes on a cluster

You can disable autoscaling on worker nodes to increase or decrease the number of nodes available by editing the machine pool definition for an existing cluster.

You can disable autoscaling on a cluster using OpenShift Cluster Manager console or the Red Hat OpenShift Service on AWS CLI.

NOTE

Additionally, you can configure autoscaling on the default machine pool when you create the cluster using interactive mode.

Disabling autoscaling nodes in an existing cluster using Red Hat OpenShift Cluster Manager
Disable autoscaling for worker nodes in the machine pool definition from OpenShift Cluster Manager console.

Procedure

1. From OpenShift Cluster Manager Hybrid Cloud Console , navigate to the **Clusters** page and select the cluster with autoscaling that must be disabled.
2. On the selected cluster, select the **Machine pools** tab.
3. Click the Options menu  at the end of the machine pool with autoscaling and select **Scale**.
4. On the "Edit node count" dialog, deselect the **Enable autoscaling** checkbox.
5. Select **Apply** to save these changes and disable autoscaling from the cluster.

Disabling autoscaling nodes in an existing cluster using the rosa CLI
Disable autoscaling for worker nodes in the machine pool definition.

Procedure

1. Enter the following command:

```
$ rosa edit machinepool --cluster=<cluster_name> <machinepool_ID> --enable-autoscaling=false --replicas=<number>
```

Example

Disable autoscaling on the **default** machine pool on a cluster named **mycluster**:

```
$ rosa edit machinepool --cluster=mycluster default --enable-autoscaling=false --replicas=3
```

2.4.3. Additional resources

- About machinepools
- Managing worker nodes
- Managing objects with the rosa CLI