Red Hat OpenShift Service on AWS 4

Setting up accounts and clusters using AWS security token service (STS)

Setting up clusters and accounts using AWS security token service (STS)
Setting up accounts and clusters using AWS security token service (STS)
Abstract

This document provides information on how to get started with Amazon Managed Red Hat OpenShift.
# Table of Contents

CHAPTER 1. GETTING STARTED USING STS WORKFLOW ......................................................... 4
  1.1. ADDITIONAL RESOURCES ................................................................................. 4

CHAPTER 2. AWS PREREQUISITES FOR ROSA WITH STS ............................................... 5
  2.1. DEPLOYMENT PREREQUISITES ......................................................................... 5
  2.2. CUSTOMER REQUIREMENTS WHEN USING STS FOR DEPLOYMENT .................... 5
    2.2.1. Account ........................................................................................................ 5
    2.2.2. Access requirements ...................................................................................... 5
    2.2.3. Support requirements .................................................................................... 6
    2.2.4. Security requirements .................................................................................... 6
  2.3. RED HAT MANAGED IAM REFERENCES FOR AWS ............................................. 6
  2.4. PROVISIONED AWS INFRASTRUCTURE ................................................................ 6
    2.4.1. EC2 instances ............................................................................................... 6
    2.4.2. Elastic Block Storage storage ....................................................................... 7
    2.4.3. Elastic load balancers .................................................................................. 7
    2.4.4. S3 storage ...................................................................................................... 7
    2.4.5. VPC ................................................................................................................ 7
    2.4.6. Security groups .............................................................................................. 8
  2.5. NEXT STEPS ....................................................................................................... 8
  2.6. ADDITIONAL RESOURCES ................................................................................ 8

CHAPTER 3. REQUIRED AWS SERVICE QUOTAS .............................................................. 9
  3.1. REQUIRED AWS SERVICE QUOTAS .................................................................. 9
  3.2. NEXT STEPS ..................................................................................................... 10

CHAPTER 4. SETTING UP THE ENVIRONMENT FOR USING STS ......................................... 11
  4.1. SETTING UP THE ENVIRONMENT FOR STS ....................................................... 11
  4.2. NEXT STEPS ................................................................................................... 15
  4.3. ADDITIONAL RESOURCES .............................................................................. 15

CHAPTER 5. CREATING IAM AND OIDC ROLES FOR STS .................................................. 16
  5.1. CREATING IAM ROLES AND POLICIES FOR STS IN ROSA ............................... 16
    5.1.1. Installer access role and policy ...................................................................... 16
    5.1.2. Control plane node instance profile role ...................................................... 20
    5.1.3. Worker node instance profile role .................................................................. 22
    5.1.4. STS support role .......................................................................................... 23
  5.2. NEXT STEPS ..................................................................................................... 27

CHAPTER 6. CREATING A CLUSTER FOR USING STS ......................................................... 28
  6.1. CREATING YOUR CLUSTER USING STS ........................................................... 28
  6.2. NEXT STEPS ................................................................................................... 32
  6.3. ADDITIONAL RESOURCES .............................................................................. 32

CHAPTER 7. CONFIGURING IDENTITY PROVIDERS FOR STS ........................................... 33
  7.1. UNDERSTANDING IDENTITY PROVIDERS ....................................................... 33
    7.1.1. Supported identity providers ......................................................................... 33
    7.1.2. Identity provider parameters ........................................................................ 33
  7.2. CONFIGURING A GITHUB IDENTITY PROVIDER .............................................. 34
  7.3. CONFIGURING A GITLAB IDENTITY PROVIDER .............................................. 36
  7.4. CONFIGURING A GOOGLE IDENTITY PROVIDER ............................................ 37
  7.5. CONFIGURING A LDAP IDENTITY PROVIDER ................................................ 38
  7.6. CONFIGURING AN OPENID IDENTITY PROVIDER .......................................... 40
  7.7. NEXT STEP ....................................................................................................... 41
CHAPTER 8. ACCESSING A ROSA CLUSTER .......................................................... 42
  8.1. ACCESSING YOUR CLUSTER QUICKLY ........................................... 42
  8.2. ACCESSING YOUR CLUSTER WITH AN IDP ACCOUNT ..................... 43
  8.3. GRANTING CLUSTER-ADMIN ACCESS ........................................... 45
  8.4. GRANTING DEDICATED-ADMIN ACCESS ....................................... 46

CHAPTER 9. DELETING ACCESS TO A ROSA CLUSTER .................................... 48
  9.1. REVOKING DEDICATED-ADMIN ACCESS ....................................... 48
  9.2. REVOKING CLUSTER-ADMIN ACCESS ........................................... 48

CHAPTER 10. DELETING A ROSA CLUSTER .................................................. 50
  10.1. DELETING A CLUSTER ............................................................... 50
CHAPTER 1. GETTING STARTED USING STS WORKFLOW

Follow this workflow to set up and access Red Hat OpenShift Service on AWS (ROSA) clusters using AWS security token service (STS).

1. Complete the AWS prerequisites for ROSA with STS.
2. Review the required AWS service quotas.
3. Set up the environment and install ROSA using STS.
4. Create IAM roles and policies for STS.
5. Create cluster using STS.
6. Configure identity providers.

1.1. ADDITIONAL RESOURCES

- Deleting a cluster
- Deleting access to a cluster
CHAPTER 2. AWS PREREQUISITES FOR ROSA WITH STS

Red Hat OpenShift Service on AWS (ROSA) provides a model that allows Red Hat to deploy clusters into a customer’s existing Amazon Web Service (AWS) account.

Ensure that the following AWS prerequisites are met before installing ROSA with STS.

2.1. DEPLOYMENT PREREQUISITES

To deploy Red Hat OpenShift Service on AWS (ROSA) into your existing Amazon Web Services (AWS) account, Red Hat requires that several prerequisites are met.

Red Hat recommends the usage of AWS Organizations to manage multiple AWS accounts. The AWS Organizations, managed by the customer, host multiple AWS accounts. There is a root account in the organization that all accounts will refer to in the account hierarchy.

It is a best practice for the ROSA cluster to be hosted in an AWS account within an AWS Organizational Unit. A Service Control Policy (SCP) is created and applied to the AWS Organizational Unit that manages what services the AWS sub-accounts are permitted to access. The SCP applies only to available permissions within a single AWS account for all AWS sub-accounts within the Organizational Unit. It is also possible to apply a SCP to a single AWS account. All other accounts in the customer’s AWS Organizations are managed in whatever manner the customer requires. Red Hat Site Reliability Engineers (SRE) will not have any control over SCPs within AWS Organizations.

2.2. CUSTOMER REQUIREMENTS WHEN USING STS FOR DEPLOYMENT

The following prerequisites must be complete before you deploy Red Hat OpenShift Service on AWS (ROSA) clusters using STS.

2.2.1. Account

- The customer ensures that the AWS limits are sufficient to support Red Hat OpenShift Service on AWS provisioned within the customer’s AWS account.

- If SCP policies are applied and enforced, these policies must not be more restrictive than the roles and policies required by the cluster.

- The customer’s AWS account should not be transferable to Red Hat.

- The customer should not impose additional AWS usage restrictions beyond the defined roles and policies on Red Hat activities. Imposing restrictions will severely hinder Red Hat’s ability to respond to incidents.

- The customer may deploy native AWS services within the same AWS account.

NOTE

Customers are encouraged, but not mandated, to deploy resources in a Virtual Private Cloud (VPC) separate from the VPC hosting Red Hat OpenShift Service on AWS and other Red Hat supported services.

2.2.2. Access requirements
Red Hat must have AWS console access to the customer-provided AWS account. This access is protected and managed by Red Hat.

The customer must not utilize the AWS account to elevate their permissions within the Red Hat OpenShift Service on AWS cluster.

Actions available in the rosa CLI utility or OpenShift Cluster Manager (OCM) console must not be directly performed in the customer’s AWS account.

2.2.3. Support requirements

- Red Hat recommends that the customer have at least Business Support from AWS.
- Red Hat may have permission from the customer to request AWS support on their behalf.
- Red Hat may have permission from the customer to request AWS resource limit increases on the customer’s account.
- Red Hat manages the restrictions, limitations, expectations, and defaults for all Red Hat OpenShift Service on AWS clusters in the same manner, unless otherwise specified in this requirements section.

2.2.4. Security requirements

- Volume snapshots will remain within the customer’s AWS account and customer-specified region.
- Red Hat must have ingress access to EC2 hosts and the API server from allow-listed IP addresses.
- Red Hat must have egress allowed to the documented domains.

2.3. RED HAT MANAGED IAM REFERENCES FOR AWS

With the STS deployment model, Red Hat is no longer responsible for creating and managing Amazon Web Services (AWS) IAM policies, IAM users, or IAM roles.

2.4. PROVISIONED AWS INFRASTRUCTURE

This is an overview of the provisioned Amazon Web Services (AWS) components on a deployed Red Hat OpenShift Service on AWS (ROSA) cluster. For a more detailed listing of all provisioned AWS components, see the OpenShift Container Platform documentation.

2.4.1. EC2 instances

AWS EC2 instances are required for deploying the control plane and data plane functions of ROSA in the AWS public cloud.

Instance types can vary for control plane and infrastructure nodes, depending on the worker node count.

- Three m5.xlarge minimum (control plane nodes)
- Two r5.xlarge minimum (infrastructure nodes)
2.4.2. Elastic Block Storage storage

Amazon EBS block storage is used for both local node storage and persistent volume storage.

Volume requirements for each EC2 instance:

- **Control Plane Volume**
  - Size: 350GB
  - Type: io1
  - Input/Output Operations Per Second: 1000

- **Infrastructure Volume**
  - Size: 300GB
  - Type: gp2
  - Input/Output Operations Per Second: 100

- **Worker Volume**
  - Size: 300GB
  - Type: gp2
  - Input/Output Operations Per Second: 100

2.4.3. Elastic load balancers

Up to two Network Elastic Load Balancers (ELBs) for API and up to two Classic ELBs for application router. For more information, see the ELB documentation for AWS.

2.4.4. S3 storage

The image registry and Elastic Block Store (EBS) volume snapshots are backed by AWS S3 storage. Pruning of resources is performed regularly to optimize S3 usage and cluster performance.

**NOTE**

Two buckets are required with a typical size of 2TB each.

2.4.5. VPC

Customers should expect to see one VPC per cluster. Additionally, the VPC will need the following configurations:

- **Subnets**: Two subnets for a cluster with a single availability zone, or six subnets for a cluster with multiple availability zones.

- **Router tables**: One router table per private subnet, and one additional table per cluster.
2.4.6. Security groups

AWS security groups provide security at the protocol and port access level; they are associated with EC2 instances and Elastic Load Balancers. Each security group contains a set of rules that filter traffic coming in and out of an EC2 instance. You must ensure the ports required for the OpenShift installation are open on your network and configured to allow access between hosts.

<table>
<thead>
<tr>
<th>Group</th>
<th>Type</th>
<th>IP Protocol</th>
<th>Port range</th>
</tr>
</thead>
<tbody>
<tr>
<td>MasterSecurityGroup</td>
<td>AWS::EC2::Security Group</td>
<td>icmp</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>6443</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>22623</td>
</tr>
<tr>
<td>WorkerSecurityGroup</td>
<td>AWS::EC2::Security Group</td>
<td>icmp</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>22</td>
</tr>
<tr>
<td>BootstrapSecurityGroup</td>
<td>AWS::EC2::Security Group</td>
<td>tcp</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>19531</td>
</tr>
</tbody>
</table>

2.5. NEXT STEPS

Review the required AWS service quotas

2.6. ADDITIONAL RESOURCES

- See SRE access to all Red Hat OpenShift Service on AWS clusters for information about how Red Hat site reliability engineering accesses ROSA clusters.
CHAPTER 3. REQUIRED AWS SERVICE QUOTAS

Review this list of the required Amazon Web Service (AWS) service quotas that are required to run an Red Hat OpenShift Service on AWS cluster.

3.1. REQUIRED AWS SERVICE QUOTAS

The table below describes the AWS service quotas and levels required to create and run an Red Hat OpenShift Service on AWS cluster.

NOTE

The AWS SDK allows ROSA to check quotas, but the AWS SDK calculation does not include your existing usage. Therefore, it is possible that the quota check can pass in the AWS SDK yet the cluster creation can fail. To fix this issue, increase your quota.

If you need to modify or increase a specific quota, see Amazon’s documentation on requesting a quota increase.

<table>
<thead>
<tr>
<th>Quota name</th>
<th>Service code</th>
<th>Quota code</th>
<th>Minimum required value</th>
<th>Recommended value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of EIPs - VPC EIPS</td>
<td>ec2</td>
<td>L-0263D0A3</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Running On-Demand Standard (A, C, D, H, I, M, R, T, Z) instances</td>
<td>ec2</td>
<td>L-1216C47A</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>VPCs per Region</td>
<td>vpc</td>
<td>L-F678F1CE</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Internet gateways per Region</td>
<td>vpc</td>
<td>L-A4707A72</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Network interfaces per Region</td>
<td>vpc</td>
<td>L-DF5E4CA3</td>
<td>5,000</td>
<td>5,000</td>
</tr>
<tr>
<td>General Purpose SSD (gp2) volume storage</td>
<td>ebs</td>
<td>L-D18FCD1D</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td>Number of EBS snapshots</td>
<td>ebs</td>
<td>L-309BACF6</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td>Provisioned IOPS</td>
<td>ebs</td>
<td>L-B3A130E6</td>
<td>300,000</td>
<td>300,000</td>
</tr>
<tr>
<td>Provisioned IOPS SSD (io1) volume storage</td>
<td>ebs</td>
<td>L-FD252861</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td>Quota name</td>
<td>Service code</td>
<td>Quota code</td>
<td>Minimum required value</td>
<td>Recommended value</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>--------------------</td>
<td>------------</td>
<td>------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Application Load Balancers per Region</td>
<td>elasticloadbalancing</td>
<td>L-53DA6B97</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>Classic Load Balancers per Region</td>
<td>elasticloadbalancing</td>
<td>L-E9E9831D</td>
<td>20</td>
<td>20</td>
</tr>
</tbody>
</table>

### 3.2. NEXT STEPS

- Set up the environment and install ROSA
CHAPTER 4. SETTING UP THE ENVIRONMENT FOR USING STS

After you meet the AWS prerequisites, set up your environment and install Red Hat OpenShift Service on AWS (ROSA).

4.1. SETTING UP THE ENVIRONMENT FOR STS

Complete the following steps to set up your environment before creating your cluster using AWS security token Service (STS).

Prerequisites

- Review and complete the deployment prerequisites and policies.
- Create a Red Hat account, if you do not already have one. Then, check your email for a verification link. You will need these credentials to install ROSA.

Procedure

1. Log in to the Amazon Web Services (AWS) account that you want to use.
   It is recommended to use a dedicated AWS account to run production clusters. If you are using AWS Organizations, you can use an AWS account within your organization or create a new one.

   If you are using AWS Organizations and you need to have a Service Control Policy (SCP) applied to the AWS account you plan to use, these policies must not be more restrictive than the roles and policies required by the cluster.

2. Enable the ROSA service in the AWS Console.
   a. Sign in to your AWS account.
   b. To enable ROSA, go to the ROSA service and select Enable OpenShift.

3. Install and configure the AWS CLI.
   a. Follow the AWS command-line interface documentation to install and configure the AWS CLI for your operating system.
      Specify the correct `aws_access_key_id` and `aws_secret_access_key` in the `.aws/credentials` file. See AWS Configuration basics in the AWS documentation.
   b. Set a default AWS region.

      NOTE
      You can use the environment variable to set the default AWS region.

      The ROSA service evaluates regions in the following priority order:
      i. The region specified when running a `rosa` command with the `--region` flag.
      ii. The region set in the `AWS_DEFAULT_REGION` environment variable. See Environment variables to configure the AWS CLI in the AWS documentation.
iii. The default region set in your AWS configuration file. See Quick configuration with aws configure in the AWS documentation.

c. Optional: Configure your AWS CLI settings and credentials by using an AWS named profile. rosa evaluates AWS named profiles in the following priority order:

i. The profile specified when running a rosa command with the --profile flag.

ii. The profile set in the AWS_PROFILE environment variable. See Named profiles in the AWS documentation.

d. Verify the AWS CLI is installed and configured correctly by running the following command to query the AWS API:

```
$ aws sts get-caller-identity
```

4. Install rosa, the Red Hat OpenShift Service on AWS command-line interface (CLI) version 1.0.8 or greater.

   a. Download the latest release of the rosa CLI for your operating system.

   b. Optional: Rename the file you downloaded to rosa and make the file executable. This documentation uses rosa to refer to the executable file.

      ```
      $ chmod +x rosa
      ```

   c. Optional: Add rosa to your path.

      ```
      $ mv rosa /usr/local/bin/rosa
      ```

   d. Enter the following command to verify your installation:

      ```
      $ rosa
      ```

Example output

Command line tool for ROSA.

Usage:
   rosa [command]

Available Commands:
   completion Generates bash completion scripts
   create Create a resource from stdin
   delete Delete a specific resource
   describe Show details of a specific resource
   edit Edit a specific resource
   help Help about any command
   init Applies templates to support Managed OpenShift on AWS clusters
   list List all resources of a specific type
   login Log in to your Red Hat account
   logout Log out
   logs Show logs of a specific resource
   verify Verify resources are configured correctly for cluster install
   version Prints the version of the tool
Optional: You can run the `rosa completion` command to generate a bash completion file.

```
$ rosa completion > /etc/bash_completion.d/rosa
```

Add this file to the correct location for your operating system. For example, on a Linux machine, run the following command to enable `rosa` bash completion:

```
$ source /etc/bash_completion.d/rosa
```

5. Download and install the OpenShift cloud credential operator tool.

```
$ git clone https://github.com/openshift/cloud-credential-operator.git
$ cd cloud-credential-operator/cmd/ccocctl
$ go build .
$ mv ccocctl /usr/local/bin/ccocctl
```

6. Log in to your Red Hat account with the `rosa` CLI.

   a. Enter the following command.

```
$ rosa login
```

   b. Replace `<my_offline_access_token>` with your token.

   **Example output**

```
To login to your Red Hat account, get an offline access token at https://cloud.redhat.com/openshift/token/rosa
? Copy the token and paste it here: <my-offline-access-token>
```

   **Example output continued**

```
I: Logged in as '<rh-rosa-user>' on 'https://api.openshift.com'
```

7. Verify that your AWS account has the necessary quota to deploy an Red Hat OpenShift Service on AWS cluster.

```
$ rosa verify quota [--region=<region>]
```
NOTE

Sometimes your AWS quota varies by region. If you receive any errors, try a different region.

If you need to increase your quota, go to your AWS console, and request a quota increase for the service that failed.

After both the permissions and quota checks pass, proceed to the next step.

8. Prepare your AWS account for cluster deployment:
   a. Run the following command to verify your Red Hat and AWS credentials are setup correctly. Check that your AWS Account ID, Default Region and ARN match what you expect. You can safely ignore the rows beginning with OCM for now (OCM stands for OpenShift Cluster Manager).

   Example output

   ```
   $ rosa whoami
   
   AWS Account ID:               000000000000
   AWS Default Region:           us-east-1
   AWS ARN:                      arn:aws:iam::000000000000:user/hello
   OCM API:                      https://api.openshift.com
   OCM Account ID:               1DzGldIhqEWyt8UUXQhSoWaaaaa
   OCM Account Name:             Your Name
   OCM Account Username:         you@domain.com
   OCM Account Email:            you@domain.com
   OCM Organization ID:          1HopHfA2hcmhup5gCr2uH5aaaaa
   OCM Organization Name:        Red Hat
   OCM Organization External ID: 0000000
   ```

   9. Install the OpenShift CLI (oc), version 4.7.9 or greater, from the ROSA (rosa) CLI.
      a. Enter this command to download the latest version of the oc CLI:

      ```
      $ rosa download openshift-client
      ```
      b. After downloading the oc CLI, unzip it and add it to your path.
      c. Enter this command to verify that the oc CLI is installed correctly:

      ```
      $ rosa verify openshift-client
      ```

   10. Optional: If you use a permissions boundary policy, gather the permissions boundary policy Amazon resource name (ARN) from your AWS account.
       Set the permissions boundary ARN as an environment variable:
$ export permissions_boundary_arn=<ARN from AWS account>

11. You can bring your own virtual private cloud (VPC) or let the installer create a VPC for you.
   a. Optional: If you are using your own Virtual Private Cloud (VPC) Gather subnet IDs for the VPC you will be installing into, and export them into a comma-separated list.
   
   $ export subnet_ids=<subnet-0abcdefg,subnet-000112233>

12. Set the rest of the static environment variables.

   $ export version=<4.7.11_or_greater> \
      name=<cluster name> \
      aws_account_id=<AWS Account ID> \
      region=<region> \
      AWS_PAGER="" # for v2 of the aws cli, unless you want commands to show you the output in `less` by default

Create roles

After completing these steps, you are ready to set up IAM and OIDC access-based roles.

4.2. NEXT STEPS

- Create IAM roles and policies for using STS

4.3. ADDITIONAL RESOURCES

- AWS Prerequisites
  
- Required AWS service quotas and increase requests
CHAPTER 5. CREATING IAM AND OIDC ROLES FOR STS

After you set up your environment for STS with Red Hat OpenShift Service on AWS (ROSA), create IAM and OIDC access-based roles.

5.1. CREATING IAM ROLES AND POLICIES FOR STS IN ROSA

When using STS, set up IAM and OIDC access-based roles before creating your cluster.

Prerequisites

- Review and complete the deployment prerequisites and policies.
- Set up the environment for STS.

5.1.1. Installer access role and policy

This role is used to manage the installation and deletion of clusters that use STS.

Procedure

1. Write the trust policy for the `ManagedOpenShift-IAM-Role` installer access role to the `ManagedOpenShift_IAM_Role.json` file.

   Example 5.1. Command to write the trust policy

   ```bash
   $ cat << EOM > ManagedOpenShift_IAM_Role.json
   {
   "Version": "2012-10-17",
   "Statement": [
   {
   "Effect": "Allow",
   "Principal": {
   "AWS": [
   "arn:aws:iam::710019948333:role/RH-Managed-OpenShift-Installer"
   ]
   },
   "Action": "sts:AssumeRole"
   }
   ]
   }
   EOM
   ```

   $ aws iam create-role --role-name ManagedOpenShift-IAM-Role --assume-role-policy-document file://ManagedOpenShift_IAM_Role.json

2. Create the `ManagedOpenShift-IAM-Role` role with the trust policy using the AWS CLI.
NOTE

If relying on a permissions boundary ARN, use the following `aws iam create-role` command instead of the previous command.

```bash
$ aws iam create-role \
    --role-name ManagedOpenShift-IAM-Role \
    --assume-role-policy-document file://ManagedOpenShift_IAM_Role.json \
    --permissions-boundary ${permissions_boundary_arn}
```

3. To create the permissions policy document, write the `ManagedOpenShift-IAM-Role-Policy` inline policy to the `ManagedOpenShift_IAM_Role_Policy.json` file.

Example 5.2. Command to create the permissions policy document

```bash
$ cat << EOM > ManagedOpenShift_IAM_Role_Policy.json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "autoscaling:DescribeAutoScalingGroups",
                "ec2:AllocateAddress",
                "ec2:AssociateAddress",
                "ec2:AssociateDhcpOptions",
                "ec2:AssociateRouteTable",
                "ec2:AttachInternetGateway",
                "ec2:AttachNetworkInterface",
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CopyImage",
                "ec2:CreateDhcpOptions",
                "ec2:CreateInternetGateway",
                "ec2:CreateNatGateway",
                "ec2:CreateNetworkInterface",
                "ec2:CreateRoute",
                "ec2:CreateRouteTable",
                "ec2:CreateSecurityGroup",
                "ec2:CreateSubnet",
                "ec2:CreateTags",
                "ec2:CreateVolume",
                "ec2:CreateVpc",
                "ec2:CreateVpcEndpoint",
                "ec2:DeleteDhcpOptions",
                "ec2:DeleteInternetGateway",
                "ec2:DeleteNatGateway",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteRoute",
                "ec2:DeleteRouteTable",
                "ec2:DeleteSecurityGroup",
                "ec2:DeleteSnapshot",
                "ec2:DeleteSubnet",
                "ec2:DeleteTags",
                "ec2:DeleteVolume",
                "ec2:DeleteVpc",
```

```bash
EOM
```
Red Hat OpenShift Service on AWS 4 Setting up accounts and clusters using AWS security token service (STS)

"ec2:DeleteVpcEndpoints",
"ec2:DeregisterImage",
"ec2:DescribeAccountAttributes",
"ec2:DescribeAddresses",
"ec2:DescribeAvailabilityZones",
"ec2:DescribeDhcpOptions",
"ec2:DescribeImages",
"ec2:DescribeInstanceAttribute",
"ec2:DescribeInstanceCreditSpecifications",
"ec2:DescribeInstances",
"ec2:DescribeInstanceStatus",
"ec2:DescribeInstanceTypes",
"ec2:DescribeInternetGateways",
"ec2:DescribeKeyPairs",
"ec2:DescribeNatGateways",
"ec2:DescribeNetworkAcls",
"ec2:DescribeNetworkInterfaces",
"ec2:DescribePrefixLists",
"ec2:DescribeRegions",
"ec2:DescribeReservedInstancesOfferings",
"ec2:DescribeRouteTables",
"ec2:DescribeSecurityGroups",
"ec2:DescribeSubnets",
"ec2:DescribeTags",
"ec2:DescribeVolumes",
"ec2:DescribeVpcAttribute",
"ec2:DescribeVpcClassicLink",
"ec2:DescribeVpcClassicLinkDnsSupport",
"ec2:DescribeVpcEndpoints",
"ec2:DescribeVpcs",
"ec2:DetachInternetGateway",
"ec2:DisassociateRouteTable",
"ec2:GetEbsDefaultKmsKeyId",
"ec2:ModifyInstanceAttribute",
"ec2:ModifyNetworkInterfaceAttribute",
"ec2:ModifySubnetAttribute",
"ec2:ModifyVpcAttribute",
"ec2:ReleaseAddress",
"ec2:ReplaceRouteTableAssociation",
"ec2:RevokeSecurityGroupEgress",
"ec2:RevokeSecurityGroupIngress",
"ec2:RunInstances",
"ec2:TerminateInstances",
"elasticloadbalancing:AddTags",
"elasticloadbalancing:ApplySecurityGroupsToLoadBalancer",
"elasticloadbalancing:AttachLoadBalancerToSubnets",
"elasticloadbalancing:ConfigureHealthCheck",
"elasticloadbalancing:CreateListener",
"elasticloadbalancing:CreateLoadBalancer",
"elasticloadbalancing:CreateLoadBalancerListeners",
"elasticloadbalancing:CreateTargetGroup",
"elasticloadbalancing:DeleteLoadBalancer",
"elasticloadbalancing:DeleteTargetGroup",
"elasticloadbalancing:DeregisterInstancesFromLoadBalancer",
"elasticloadbalancing:DeregisterTargets",
"elasticloadbalancing:DescribeInstanceHealth",
"elasticloadbalancing:DescribeListeners",
"elasticloadbalancing:DescribeLoadBalancerAttributes",
"elasticloadbalancing:DescribeLoadBalancers",
"elasticloadbalancing:DescribeTags",
"elasticloadbalancing:DescribeTargetGroupAttributes",
"elasticloadbalancing:DescribeTargetGroups",
"elasticloadbalancing:DescribeTargetHealth",
"elasticloadbalancing:ModifyLoadBalancerAttributes",
"elasticloadbalancing:ModifyTargetGroup",
"elasticloadbalancing:ModifyTargetGroupAttributes",
"elasticloadbalancing:RegisterInstancesWithLoadBalancer",
"elasticloadbalancing:RegisterTargets",
"elasticloadbalancing:SetLoadBalancerPoliciesOfListener",
"iam:AddRoleToInstanceProfile",
"iam:CreateInstanceProfile",
"iam:DeleteInstanceProfile",
"iam:GetInstanceProfile",
"iam:GetRole",
"iam:GetRolePolicy",
"iam:GetUser",
"iam:ListAttachedRolePolicies",
"iam:ListInstanceProfiles",
"iam:ListInstanceProfilesForRole",
"iam:ListRolePolicies",
"iam:ListRoles",
"iam:ListUserPolicies",
"iam:ListUsers",
"iam:PassRole",
"iam:RemoveRoleFromInstanceProfile",
"iam:SimulatePrincipalPolicy",
"iam:TagRole",
"iam:UntagRole",
"route53:ChangeResourceRecordSets",
"route53:ChangeTagsForResource",
"route53:CreateHostedZone",
"route53:DeleteHostedZone",
"route53:GetChange",
"route53:GetHostedZone",
"route53:ListHostedZones",
"route53:ListHostedZonesByName",
"route53:ListResourceRecordSets",
"route53:ListTagsForResource",
"route53:UpdateHostedZoneComment",
"s3:CreateBucket",
"s3:DeleteBucket",
"s3:DeleteObject",
"s3:GetAccelerateConfiguration",
"s3:GetBucketAcl",
"s3:GetBucketCORS",
"s3:GetBucketLocation",
"s3:GetBucketLogging",
"s3:GetBucketObjectLockConfiguration",
"s3:GetBucketRequestPayment",
"s3:GetBucketTagging",
"s3:GetBucketVersioning",
"s3:GetBucketWebsite"
Attach the inline policy in the ManagedOpenShift_IAM_Role_Policy.json file to the ManagedOpenShift-IAM-Role instance profile role.

![Image](Attach_Inline_Policy.jpg)

4. Attach the inline policy in the ManagedOpenShift_IAM_Role_Policy.json file to the ManagedOpenShift-IAM-Role instance profile role.

```bash
$ aws iam put-role-policy --role-name ManagedOpenShift-IAM-Role --policy-name ManagedOpenShift-IAM-Role-Policy --policy-document file://ManagedOpenShift_IAM_Role_Policy.json
```

5.1.2. Control plane node instance profile role

**Procedure**

1. Write the trust policy for the ManagedOpenShift-ControlPlane-Role control plane node instance profile role to the ManagedOpenShift_ControlPlane_Role.json file.

   **Example 5.3. Command to write the trust policy**

   ```bash
   $ cat << EOM > ManagedOpenShift_ControlPlane_Role.json
   {
   "Version": "2012-10-17",
   "Statement": [
   {
   "Effect": "Allow",
   "Principal": {
   "Service": "ec2.amazonaws.com"
   },
   "Action": "sts:AssumeRole"
   }
   ]
   "Resource": "*"
   }
   EOM
   ```
2. Create the **ManagedOpenShift-ControlPlane-Role** role with the trust policy using the AWS CLI.

```bash
$ aws iam create-role --role-name ManagedOpenShift-ControlPlane-Role --assume-role-policy-document file://ManagedOpenShift_ControlPlane_Role.json
```

**NOTE**

If relying on a permissions boundary ARN, use the following `aws iam create-role` command instead of the previous command.

```bash
$ aws iam create-role
   --role-name ManagedOpenShift-ControlPlane-Role
   --assume-role-policy-document
   file://ManagedOpenShift_ControlPlane_Role.json
   --permissions-boundary ${permissions_boundary_arn}
```

3. To create the permissions policy document, write the **ManagedOpenShift-ControlPlane-Role-Policy** inline policy to the `ManagedOpenShift_ControlPlane_Role_Policy.json` file.

**Example 5.4. Command to create the permissions policy document**

```bash
$ cat << EOM > ManagedOpenShift_ControlPlane_Role_Policy.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:AttachVolume",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:CreateSecurityGroup",
        "ec2:CreateTags",
        "ec2:CreateVolume",
        "ec2:DeleteSecurityGroup",
        "ec2:DeleteVolume",
        "ec2:Describe**",
        "ec2:DetachVolume",
        "ec2:ModifyInstanceAttribute",
        "ec2:ModifyVolume",
        "ec2:RevokeSecurityGroupIngress",
        "elasticloadbalancing:AddTags",
        "elasticloadbalancing:AttachLoadBalancerToSubnets",
        "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer",
        "elasticloadbalancing:CreateListener",
        "elasticloadbalancing:CreateLoadBalancer",
        "elasticloadbalancing:CreateLoadBalancerPolicy",
        "elasticloadbalancing:CreateLoadBalancerListeners",
        "elasticloadbalancing:CreateTargetGroup",
```
4. Attach the inline policy in the `ManagedOpenShift_ControlPlane_Role_Policy.json` file to the `ManagedOpenShift-ControlPlane-Role` instance profile role.

```bash
$ aws iam put-role-policy --role-name ManagedOpenShift-ControlPlane-Role --policy-name ManagedOpenShift-ControlPlane-Role-Policy --policy-document file://ManagedOpenShift_ControlPlane_Role_Policy.json
```

5.1.3. Worker node instance profile role

1. Write the trust policy for the `ManagedOpenShift-Worker-Role` access instance profile role to the `ManagedOpenShift_Worker_Role.json` file.

Example 5.5. Command to write the trust policy

```bash
$ cat <<EOM > ManagedOpenShift_Worker_Role.json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ],
    "Resource": "*"
}
}
EOM
```
2. Create the **ManagedOpenShift-Worker-Role** role with the trust policy using the AWS CLI.

   
   ```bash
   $ aws iam create-role --role-name ManagedOpenShift-Worker-Role --assume-role-policy-document file://ManagedOpenShift_Worker_Role.json
   ``

   **NOTE**

   If relying on a permissions boundary ARN, use the following `aws iam create-role` command instead of the previous command.

   ```bash
   $ aws iam create-role \
   --role-name ManagedOpenShift-Worker-Role \
   --assume-role-policy-document file://ManagedOpenShift_Worker_Role.json \
   --permissions-boundary ${permissions_boundary_arn}
   ```

3. Write the **ManagedOpenShift-Worker-Role-Policy** policy to the **ManagedOpenShift_Worker_Role_Policy.json** file to create the permissions policy document.

   **Example 5.6. Command to create the permissions policy document**

   ```bash
   $ cat << EOM > ManagedOpenShift_Worker_Role_Policy.json
   {
   "Version": "2012-10-17",
   "Statement": [
   {"Effect": "Allow",
   "Action": ["ec2:DescribeInstances",
   "ec2:DescribeRegions"],
   "Resource": "*"}
   ]
   }
   EOM
   ```

4. Attach the **ManagedOpenShift_Worker_Role_Policy.json** file to the **ManagedOpenShift-Worker-Role** instance profile role.

   ```bash
   $ aws iam put-role-policy --role-name ManagedOpenShift-Worker-Role --policy-name ManagedOpenShift-Worker-Role-Policy --policy-document file://ManagedOpenShift_Worker_Role_Policy.json
   ``

5.1.4. STS support role

The STS support role is designed to give Red Hat site reliability engineering (SRE) read-only access to support a given cluster and troubleshoot issues.

1. Write the trust policy for the **RH-Technical-Support-Access** instance profile role to the **RH_Support_Role.json** file.

   **Example 5.7. Command to write the trust policy**

   ```bash
   ```
Create the ManagedOpenShift-Support-Role role with the trust policy using the AWS CLI.

```
$ cat << EOM > RH_Support_Role.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": [
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOM
```

2. Create the **ManagedOpenShift-Support-Role** role with the trust policy using the AWS CLI.

```
$ aws iam create-role --role-name ManagedOpenShift-Support-Role --assume-role-policy-document file://RH_Support_Role.json
```

**NOTE**

If relying on a permissions boundary ARN, use the following `aws iam create-role` command instead of the previous command.

```
$ aws iam create-role --role-name ManagedOpenShift-Support-Role --assume-role-policy-document file://RH_Support_Role.json --permissions-boundary ${permissions_boundary_arn}
```

3. Write the **ManagedOpenShift-Support-Role** policy to the `RH_Support_Policy.json` file to create the permissions policy document.

**Example 5.8. Command to create the permissions policy document**

```
$ cat << EOM > RH_Support_Policy.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudtrail:DescribeTrails",
        "cloudtrail:LookupEvents",
        "cloudwatch:GetMetricData",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:ListMetrics",
        "ec2:CopySnapshot",
        "ec2:CreateSnapshot",
        "ec2:CreateSnapshots",
        "ec2:DescribeAccountAttributes",
```

Red Hat OpenShift Service on AWS 4 Setting up accounts and clusters using AWS security token service (STS)
"ec2:DescribeAddresses",
"ec2:DescribeAddressesAttribute",
"ec2:DescribeAggregateIdFormat",
"ec2:DescribeAvailabilityZones",
"ec2:DescribeByoipCidrs",
"ec2:DescribeCapacityReservations",
"ec2:DescribeCarrierGateways",
"ec2:DescribeClassicLinkInstances",
"ec2:DescribeClientVpnAuthorizationRules",
"ec2:DescribeClientVpnConnections",
"ec2:DescribeClientVpnEndpoints",
"ec2:DescribeClientVpnRoutes",
"ec2:DescribeClientVpnTargetNetworks",
"ec2:DescribeCoipPools",
"ec2:DescribeCustomerGateways",
"ec2:DescribeDhcpOptions",
"ec2:DescribeEgressOnlyInternetGateways",
"ec2:DescribeIamInstanceProfileAssociations",
"ec2:DescribeIdFormat",
"ec2:DescribeIdentityIdFormat",
"ec2:DescribeImageAttribute",
"ec2:DescribeImages",
"ec2:DescribeInstanceAttribute",
"ec2:DescribeInstanceStatus",
"ec2:DescribeInstanceTypeOfferings",
"ec2:DescribeInstanceTypes",
"ec2:DescribeInstances",
"ec2:DescribeInternetGateways",
"ec2:DescribeIpv6Pools",
"ec2:DescribeKeyPairs",
"ec2:DescribeLaunchTemplates",
"ec2:DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations",
"ec2:DescribeLocalGatewayRouteTableVpcAssociations",
"ec2:DescribeLocalGatewayRouteTables",
"ec2:DescribeLocalGatewayVirtualInterfaceGroups",
"ec2:DescribeLocalGatewayVirtualInterfaces",
"ec2:DescribeLocalGateways",
"ec2:DescribeNatGateways",
"ec2:DescribeNetworkAcls",
"ec2:DescribeNetworkInterfaces",
"ec2:DescribePlacementGroups",
"ec2:DescribePrefixLists",
"ec2:DescribePrincipalIdFormat",
"ec2:DescribePublicIpv4Pools",
"ec2:DescribeRegions",
"ec2:DescribeReservedInstances",
"ec2:DescribeRouteTables",
"ec2:DescribeScheduledInstances",
"ec2:DescribeSecurityGroupReferences",
"ec2:DescribeSecurityGroups",
"ec2:DescribeSnapshotAttribute",
"ec2:DescribeSnapshots",
"ec2:DescribeSpotFleetInstances",
"ec2:DescribeStaleSecurityGroups",
"ec2:DescribeSubnets",
"ec2:DescribeTags",
Create the `ManagedOpenShift-Support-Access` policy object in AWS using the AWS CLI.

```
$ aws iam create-policy --policy-name ManagedOpenShift-Support-Access --policy-document file://RH_Support_Policy.json
```

5. Attach the `ManagedOpenShift-Support-Access` policy to the `ManagedOpenShift-Support-Role` role:

```
$ policy_arn=<output_of_policy_arn_from_above_command>
aws iam attach-role-policy --role-name ManagedOpenShift-Support-Role --policy-arn $policy_arn
```

Install

After completing these steps, you are ready to create a cluster.

5.2. NEXT STEPS

Create a cluster
CHAPTER 6. CREATING A CLUSTER FOR USING STS

After you set up your environment and install Red Hat OpenShift Service on AWS (ROSA), create a cluster.

6.1. CREATING YOUR CLUSTER USING STS

You can create a Red Hat OpenShift Service on AWS cluster with AWS security token service (STS) using the ROSA CLI (rosa).

Prerequisites

You have completed the installation prerequisites.

NOTE

AWS Shared VPCs are not currently supported for ROSA installations.

Procedure

When creating your cluster, use the following default options to configure your networking IP ranges. For more information when using manual mode, use `rosa create cluster --help | grep cidr`. In interactive mode, you are prompted for the settings.

- Machine CIDR: 10.0.0.0/16
- Service CIDR: 172.30.0.0/16
- Pod CIDR: 10.128.0.0/14

1. You can create a cluster by specifying the custom settings shown below or by using the interactive mode. To view other options when creating a cluster, enter `rosa create cluster --help`. Creating a cluster can take up to 40 minutes.

   NOTE

   Multiple availability zones (Multi-AZ) are recommended for production workloads. The default is a single availability zone. Use `--help` for an example of how to set this option manually or use interactive mode to be prompted for this setting.

   - To create a cluster with STS using custom settings:

     ```bash
     $ rosa create cluster 
     --cluster-name ${name} 
     --region ${region} 
     --version ${version} 
     --role-arn arn:aws:iam::${aws_account_id}:role/ManagedOpenShift-IAM-Role 
     --support-role-arn arn:aws:iam::${aws_account_id}:role/ManagedOpenShift-Support-Role 
     --master-iam-role arn:aws:iam::${aws_account_id}:role/ManagedOpenShift-ControlPlane-Role 
     --worker-iam-role arn:aws:iam::${aws_account_id}:role/ManagedOpenShift-Worker-Role 
     ```
To create a cluster with STS using interactive prompts:

```bash
$ rosa create cluster --interactive
```

2. To check the status of your cluster and retrieve your cluster ID, enter the next command. The 'State' field changes from 'pending' to 'installing' to 'ready':

```bash
$ rosa describe cluster --cluster=${name}
```

**Example output**

```
I: Creating cluster with identifier '<cluster_id>' and name '<cluster_name>'
I: To view list of clusters and their status, run `rosa list clusters`
I: Cluster '<cluster_name>' has been created.
I: Once the cluster is 'Ready' you will need to add an Identity Provider and define the
list of cluster administrators. See `rosa create idp --help` and `rosa create user --help` for more information.
I: To determine when your cluster is Ready, run `rosa describe cluster
<cluster_name>`.
```

- To create a cluster with STS using interactive prompts:

  ```bash
  $ rosa create cluster --interactive
  ```

**Example output**

```
Name:                      <cluster_name>
ID:                        <cluster_id>
External ID:                <external_id>
OpenShift Version:          <version>
Channel Group:              stable
DNS:                       *.openshiftapps.com
AWS Account:                123456789012
API URL:                    https://api.<cluster_name>.openshiftapps.com:6443
Console URL:                https://console-openshift-console.apps.<cluster_name>.openshiftapps.com
<cluster_name>.openshiftapps.com
Region:                    <region>
Multi-AZ:                   false
Nodes:
  - Master:                  3
  - Infra:                   2
  - Compute:                 2
```
If installation fails or the State field does not change to ready after 40 minutes, check the installation troubleshooting documentation for more details.

3. Set the environment variable for the **cluster_id** from the previous output:

```bash
$ export cluster_id=<cluster_id>
```

**IMPORTANT**

Before proceeding, wait for the **State** output from `rosa describe cluster` to change to pending (Waiting for OIDC configuration).

4. Complete the setup of the OIDC access-based IAM roles.

   a. Create the OIDC provider:

   ```bash
   $ thumbprint=$(openssl s_client \
   -servername rh-oidc.s3.us-east-1.amazonaws.com/${cluster_id} \n   -showcerts \n   -connect rh-oidc.s3.us-east-1.amazonaws.com:443 </dev/null 2>&1 | openssl x509 \
   -fingerprint \n   -noout | tail -n1 | sed 's/SHA1 Fingerprint=//' | sed 's://g')

   $ aws iam create-open-id-connect-provider \
   --url https://rh-oidc.s3.us-east-1.amazonaws.com/${cluster_id} \n   --client-id-list openshift sts.amazonaws.com \n   --thumbprint-list ${thumbprint}
   ```

   **NOTE**

   If the certificate changes, you will need to update the thumbprint.

   b. Generate permissions for OIDC-access-based roles Extract credential requests from the desired Red Hat OpenShift Service on AWS version:
NOTE

This action requires the OpenShift CLI (oc), version 4.7.9 or greater. You
can download the latest oc version from the ROSA (rosa) CLI.

$ rosa download openshift-client

After downloading the oc CLI, unzip it and add it to your path.

c. Create the IAM roles:

$ mkdir -p iam_assets
$ cd iam_assets

$ ccoctl aws create-iam-roles \
   --credentials-requests-dir ../credrequests/ \
   --identity-provider-arn arn:aws:iam::${aws_account_id}:oidc-provider/rh-oidc.s3.us-east-1.amazonaws.com/${cluster_id} \
   --name ManagedOpenShift \
   --region ${region} \
   --dry-run

$ for role in `find . -name "*-role.json"`
do
    policy=$(sed -e 's/05-/06-/' -e 's/role/policy/' <<< $role)
    role_name=$(grep --color=never -o "RoleName":"\"(\w|-)*"" $policy | sed "s/RoleName":"//")
    aws iam create-role --cli-input-json file://${role} sed -i.bak 's/"RoleName":"\"\"/"RoleName":"$role_name"/'
    policy_arn=$(aws iam create-policy --output json --cli-input-json file://$policy | grep Arn | awk '{print $2}' | awk -F" " '{print $2}"
    aws iam attach-role-policy --role-name $role_name --policy-arn $policy_arn
    rm $policy
    mv $policy.bak $policy
    sleep 5 # Prevents AWS Rate limiting
done
NOTE

If relying on a permissions boundary ARN, use the following `aws iam create-role` command in the previous loop:

```
$ aws iam create-role \
    --cli-input-json file://${role} \
    --permissions-boundary ${permissions_boundary_arn}
```

5. To track the progress of your cluster creation, enter this command to watch the OpenShift installer logs:

```
$ rosa logs install --cluster=${name} --watch
```

6.2. NEXT STEPS

Configure identity providers

6.3. ADDITIONAL RESOURCES

- Troubleshooting
CHAPTER 7. CONFIGURING IDENTITY PROVIDERS FOR STS

After your Red Hat OpenShift Service on AWS (ROSA) cluster is created, you must configure identity providers to determine how users log in to access the cluster.

7.1. UNDERSTANDING IDENTITY PROVIDERS

Red Hat OpenShift Service on AWS includes a built-in OAuth server. Developers and administrators obtain OAuth access tokens to authenticate themselves to the API. As an administrator, you can configure OAuth to specify an identity provider after you install your cluster. Configuring identity providers allows users to log in and access the cluster.

7.1.1. Supported identity providers

You can configure the following types of identity providers:

<table>
<thead>
<tr>
<th>Identity provider</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GitHub or GitHub Enterprise</td>
<td>Configure a <strong>github</strong> identity provider to validate user names and passwords against GitHub or GitHub Enterprise's OAuth authentication server.</td>
</tr>
<tr>
<td>GitLab</td>
<td>Configure a <strong>gitlab</strong> identity provider to use GitLab.com or any other GitLab instance as an identity provider.</td>
</tr>
<tr>
<td>Google</td>
<td>Configure a <strong>google</strong> identity provider using Google's OpenID Connect integration.</td>
</tr>
<tr>
<td>LDAP</td>
<td>Configure the <strong>ldap</strong> identity provider to validate user names and passwords against an LDAPv3 server, using simple bind authentication.</td>
</tr>
<tr>
<td>OpenID Connect</td>
<td>Configure an <strong>oidc</strong> identity provider to integrate with an OpenID Connect identity provider using an Authorization Code Flow.</td>
</tr>
</tbody>
</table>

7.1.2. Identity provider parameters

The following parameters are common to all identity providers:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The provider name is prefixed to provider user names to form an identity name.</td>
</tr>
</tbody>
</table>
Defines how new identities are mapped to users when they log in. Enter one of the following values:

**claim**
The default value. Provisions a user with the identity’s preferred user name. Fails if a user with that user name is already mapped to another identity.

**lookup**
Looks up an existing identity, user identity mapping, and user, but does not automatically provision users or identities. This allows cluster administrators to set up identities and users manually, or using an external process. Using this method requires you to manually provision users.

**generate**
Provisions a user with the identity’s preferred user name. If a user with the preferred user name is already mapped to an existing identity, a unique user name is generated. For example, `myuser2`. This method should not be used in combination with external processes that require exact matches between Red Hat OpenShift Service on AWS user names and identity provider user names, such as LDAP group sync.

**add**
Provisions a user with the identity’s preferred user name. If a user with that user name already exists, the identity is mapped to the existing user, adding to any existing identity mappings for the user. Required when multiple identity providers are configured that identify the same set of users and map to the same user names.

---

**NOTE**
When adding or changing identity providers, you can map identities from the new provider to existing users by setting the `mappingMethod` parameter to `add`.

### 7.2. CONFIGURING A GITHUB IDENTITY PROVIDER

Configure a GitHub identity provider to validate user names and passwords against GitHub or GitHub Enterprise’s OAuth authentication server and access your Red Hat OpenShift Service on AWS cluster. OAuth facilitates a token exchange flow between Red Hat OpenShift Service on AWS and GitHub or GitHub Enterprise.

**WARNING**
Configuring GitHub authentication allows users to log in to Red Hat OpenShift Service on AWS with their GitHub credentials. To prevent anyone with any GitHub user ID from logging in to your Red Hat OpenShift Service on AWS cluster, you must restrict access to only those in specific GitHub organizations or teams.

---

**Prerequisites**
• The OAuth application must be created directly within the GitHub organization settings by the GitHub organization administrator.

• GitHub organizations or teams are set up in your GitHub account.

Procedure

1. Navigate to the Clusters page and select the cluster that you need to configure identity providers for.

2. Click the Access control tab.

3. Click Add identity provider.

   NOTE

   You can also click the Add Oauth configuration link in the warning message displayed after cluster creation to configure your identity providers.

4. Select GitHub from the drop-down menu.

5. Enter a unique name for the identity provider. This name cannot be changed later.

   • An OAuth callback URL is automatically generated in the provided field. You will use this to register the GitHub application.

     https://oauth-openshift.apps.<cluster_name>.-<cluster_domain>/oauth2callback/<idp_provider_name>

     For example:

     https://oauth-openshift.apps.example-openshift-cluster.com/oauth2callback/github/

6. Register an application on GitHub.

7. Return to Red Hat OpenShift Service on AWS and select a mapping method from the drop-down menu. Claim is recommended in most cases.

8. Enter the Client ID and Client secret provided by GitHub.

9. Enter a hostname. A hostname must be entered when using a hosted instance of GitHub Enterprise.

10. Optional: You can use a certificate authority (CA) file to validate server certificates for the configured GitHub Enterprise URL. Click Browse to locate and attach a CA file to the identity provider.

11. Select Use organizations or Use teams to restrict access to a particular GitHub organization or a GitHub team.

12. Enter the name of the organization or team you would like to restrict access to. Click Add more to specify multiple organizations or teams that users can be a member of.

13. Click Confirm.
Verification

- The configured identity provider is now visible on the Access control tab of the Clusters page.

### 7.3. CONFIGURING A GITLAB IDENTITY PROVIDER

Configure a GitLab identity provider to use GitLab.com or any other GitLab instance as an identity provider.

**Prerequisites**

- If you use GitLab version 7.7.0 to 11.0, you connect using the OAuth integration. If you use GitLab version 11.1 or later, you can use OpenID Connect (OIDC) to connect instead of OAuth.

**Procedure**

1. Navigate to the Clusters page and select the cluster that you need to configure identity providers for.
2. Click the Access control tab.
3. Click Add identity provider.

   **NOTE**

   You can also click the Add Oauth configuration link in the warning message displayed after cluster creation to configure your identity providers.

4. Select GitLab from the drop-down menu.
5. Enter a unique name for the identity provider. This name cannot be changed later.

   - An OAuth callback URL is automatically generated in the provided field. You will provide this URL to GitLab.

     ```
     https://oauth-openshift.apps.<cluster_name>.<cluster_domain>/oauth2callback/<idp_provider_name>
     ``

     For example:

     ```
     https://oauth-openshift.apps.example-openshift-cluster.com/oauth2callback/gitlab/
     ```

6. Add a new application in GitLab.
7. Return to Red Hat OpenShift Service on AWS and select a mapping method from the drop-down menu. Claim is recommended in most cases.
8. Enter the Client ID and Client secret provided by GitLab.
9. Enter the URL of your GitLab provider.
10. Optional: You can use a certificate authority (CA) file to validate server certificates for the configured GitLab URL. Click Browse to locate and attach a CA file to the identity provider.
11. Click **Confirm**.

**Verification**

- The configured identity provider is now visible on the **Access control** tab of the **Clusters** page.

### 7.4. CONFIGURING A GOOGLE IDENTITY PROVIDER

Configure a Google identity provider to allow users to authenticate with their Google credentials.

**WARNING**

Using Google as an identity provider allows any Google user to authenticate to your server. You can limit authentication to members of a specific hosted domain with the **hostedDomain** configuration attribute.

**Procedure**

1. Navigate to the **Clusters** page and select the cluster that you need to configure identity providers for.

2. Click the **Access control** tab.

3. Click **Add identity provider**.

**NOTE**

You can also click the **Add Oauth configuration** link in the warning message displayed after cluster creation to configure your identity providers.

4. Select **Google** from the drop-down menu.

5. Enter a unique name for the identity provider. This name cannot be changed later.

   - An **OAuth callback URL** is automatically generated in the provided field. You will provide this URL to Google.

   ```
   https://oauth-openshift.apps.<cluster_name>.<cluster_domain>/oauth2callback/<idp_provider_name>
   ```

   For example:

   ```
   https://oauth-openshift.apps.example-openshift-cluster.com/oauth2callback/github/
   ```

6. Configure a Google identity provider using **Google’s OpenID Connect integration**.

7. Return to Red Hat OpenShift Service on AWS and select a mapping method from the drop-down menu. **Claim** is recommended in most cases.
8. Enter the Client ID of a registered Google project and the Client secret issued by Google.

9. Enter a hosted domain to restrict users to a Google Apps domain.

10. Click Confirm.

**Verification**

- The configured identity provider is now visible on the Access control tab of the Clusters page.

### 7.5. CONFIGURING A LDAP IDENTITY PROVIDER

Configure the LDAP identity provider to validate user names and passwords against an LDAPv3 server, using simple bind authentication.

**Prerequisites**

- When configuring a LDAP identity provider, you will need to enter a configured LDAP URL. The configured URL is an RFC 2255 URL, which specifies the LDAP host and search parameters to use. The syntax of the URL is:

  \[ldap://host:port/basedn?attribute?scope?filter\]

<table>
<thead>
<tr>
<th>URL component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldap</td>
<td>For regular LDAP, use the string ldap. For secure LDAP (LDAPS), use ldaps instead.</td>
</tr>
<tr>
<td>host:port</td>
<td>The name and port of the LDAP server. Defaults to localhost:389 for ldap and localhost:636 for LDAPS.</td>
</tr>
<tr>
<td>basedn</td>
<td>The DN of the branch of the directory where all searches should start from. At the very least, this must be the top of your directory tree, but it could also specify a subtree in the directory.</td>
</tr>
<tr>
<td>attribute</td>
<td>The attribute to search for. Although RFC 2255 allows a comma-separated list of attributes, only the first attribute will be used, no matter how many are provided. If no attributes are provided, the default is to use uid. It is recommended to choose an attribute that will be unique across all entries in the subtree you will be using.</td>
</tr>
<tr>
<td>scope</td>
<td>The scope of the search. Can be either one or sub. If the scope is not provided, the default is to use a scope of sub.</td>
</tr>
<tr>
<td>filter</td>
<td>A valid LDAP search filter. If not provided, defaults to (objectClass=*)</td>
</tr>
</tbody>
</table>

When doing searches, the attribute, filter, and provided user name are combined to create a search filter that looks like:

\[(&(<filter>)(<attribute>=<username>))\]
IMPORTANT

If the LDAP directory requires authentication to search, specify a **bindDN** and **bindPassword** to use to perform the entry search.

**Procedure**

1. Navigate to the **Clusters** page and select the cluster that you need to configure identity providers for.
2. Click the **Access control** tab.
3. Click **Add identity provider**.

**NOTE**

You can also click the **Add Oauth configuration** link in the warning message displayed after cluster creation to configure your identity providers.

4. Select **LDAP** from the drop-down menu.
5. Enter a unique name for the identity provider. This name cannot be changed later.
6. Select a mapping method from the drop-down menu. **Claim** is recommended in most cases.
7. Enter a **LDAP URL** to specify the LDAP search parameters to use.
8. Optional: Enter a **Bind DN** and **Bind password**.
9. Enter the attributes that will map LDAP attributes to identities.
   - Enter an **ID** attribute whose value should be used as the user ID. Click **Add more** to add multiple ID attributes.
   - Optional: Enter a **Preferred username** attribute whose value should be used as the display name. Click **Add more** to add multiple preferred username attributes.
   - Optional: Enter an **Email** attribute whose value should be used as the email address. Click **Add more** to add multiple email attributes.
10. Optional: Click **Show advanced Options** to add a certificate authority (CA) file to your LDAP identity provider to validate server certificates for the configured URL. Click **Browse** to locate and attach a **CA file** to the identity provider.
11. Optional: Under the advanced options, you can choose to make the LDAP provider **Insecure**. If you select this option, a CA file cannot be used.

**IMPORTANT**

If you are using an insecure LDAP connection (ldap:// or port 389), then you must check the **Insecure** option in the configuration wizard.

12. Click **Confirm**.

**Verification**
The configured identity provider is now visible on the Access control tab of the Clusters page.

7.6. CONFIGURING AN OPENID IDENTITY PROVIDER

Configure an OpenID identity provider to integrate with an OpenID Connect identity provider using an Authorization Code Flow.

IMPORTANT

The Authentication Operator in Red Hat OpenShift Service on AWS requires that the configured OpenID Connect identity provider implements the OpenID Connect Discovery specification.

Claims are read from the JWT id_token returned from the OpenID identity provider and, if specified, from the JSON returned by the Issuer URL.

At least one claim must be configured to use as the user’s identity.

You can also indicate which claims to use as the user’s preferred user name, display name, and email address. If multiple claims are specified, the first one with a non-empty value is used. The standard claims are:

<table>
<thead>
<tr>
<th>Claim</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>preferred_username</td>
<td>The preferred user name when provisioning a user. A shorthand name that the user wants to be referred to as, such as janedoe. Typically a value that corresponds to the user's login or username in the authentication system, such as username or email.</td>
</tr>
<tr>
<td>email</td>
<td>Email address.</td>
</tr>
<tr>
<td>name</td>
<td>Display name.</td>
</tr>
</tbody>
</table>

See the OpenID claims documentation for more information.

Prerequisites

- Before you configure OpenID Connect, check the installation prerequisites for any Red Hat product or service you want to use with your Red Hat OpenShift Service on AWS cluster.

Procedure

1. Navigate to the Clusters page and select the cluster that you need to configure identity providers for.

2. Click the Access control tab.

3. Click Add identity provider.
NOTE

You can also click the Add Oauth configuration link in the warning message displayed after cluster creation to configure your identity providers.

4. Select OpenID from the drop-down menu.

5. Enter a unique name for the identity provider. This name cannot be changed later.
   - An OAuth callback URL is automatically generated in the provided field.

```
https://oauth-openshift.apps.<cluster_name>.
<cluster_domain>/oauth2callback/<idp_provider_name>
```

For example:

```
https://oauth-openshift.apps.example-openshift-cluster.com/oauth2callback/openid/
```


7. Return to Red Hat OpenShift Service on AWS and select a mapping method from the drop-down menu. Claim is recommended in most cases.

8. Enter a Client ID and Client secret provided from OpenID.

9. Enter an Issuer URL. This is the URL that the OpenID provider asserts as the Issuer Identifier. It must use the https scheme with no URL query parameters or fragments.

10. Enter an Email attribute whose value should be used as the email address. Click Add more to add multiple email attributes.

11. Enter a Name attribute whose value should be used as the preferred username. Click Add more to add multiple preferred usernames.

12. Enter a Preferred username attribute whose value should be used as the display name. Click Add more to add multiple display names.

13. Optional: Click Show advanced Options to add a certificate authority (CA) file to your OpenID identity provider.

14. Optional: Under the advanced options, you can add Additional scopes. By default, the OpenID scope is requested.

15. Click Confirm.

Verification
   - The configured identity provider is now visible on the Access control tab of the Clusters page.

7.7. NEXT STEP

Access a cluster
CHAPTER 8. ACCESSING A ROSA CLUSTER

It is recommended that you access your Red Hat OpenShift Service on AWS (ROSA) cluster using an identity provider (IDP) account. However, the cluster administrator who created the cluster can access it using the quick access procedure.

8.1. ACCESSING YOUR CLUSTER QUICKLY

You can use this quick access procedure to log in to your cluster.

NOTE

As a best practice, access your cluster with an IDP account instead.

Procedure

1. Enter the following command:

```
$ rosa create admin --cluster=<cluster_name>
```

Example output

W: It is recommended to add an identity provider to login to this cluster. See 'rosa create idp --help' for more information.
I: Admin account has been added to cluster 'cluster_name'. It may take up to a minute for the account to become active.
I: To login, run the following command:
  oc login https://api.cluster-name.t6k4.i1.oragnization.org:6443 
  --username cluster-admin 
  --password FWGYL-2mkJI-3ZTTZ-rINns

2. Enter the `oc login` command, username, and password from the output of the previous command:

Example output

```
$ oc login https://api.cluster_name.t6k4.i1.oragnization.org:6443 
  > --username cluster-admin 
  > --password FWGYL-2mkJI-3ZTTZ-rINns
Login successful.
You have access to 77 projects, the list has been suppressed. You can list all projects with ' projects'
```

3. Using the default project, enter this `oc` command to verify that the cluster administrator access is created:

```
$ oc whoami
```

Example output

```
cluster-admin
```
8.2. ACCESSING YOUR CLUSTER WITH AN IDP ACCOUNT

To log in to your cluster, you can configure an identity provider (IDP). This procedure uses GitHub as an example IDP. To view other supported IDPs, run the `rosa create idp --help` command.

**NOTE**

Alternatively, as the user who created the cluster, you can use the quick access procedure.

**Procedure**

To access your cluster using an IDP account:

1. **Add an IDP.**
   
   a. The following command creates an IDP backed by GitHub. After running the command, follow the interactive prompts from the output to access your GitHub developer settings and configure a new OAuth application.

   
   ```
   $ rosa create idp --cluster=<cluster_name> --interactive
   ```

   b. Enter the following values:

   - Type of identity provider: `github`
   
   - Restrict to members of: `organizations` (if you do not have a GitHub Organization, you can create one now)
   
   - GitHub organizations: `rh-test-org` (enter the name of your organization)

   **Example output**

   ```
   I: Interactive mode enabled.
   Any optional fields can be left empty and a default will be selected.
   ? Type of identity provider: github
   ? Restrict to members of: organizations
   ? GitHub organizations: rh-test-org
   ? To use GitHub as an identity provider, you must first register the application:
   - Open the following URL:
     https://github.com/organizations/rh-rosa-test-cluster/settings/applications/new?
     oauth_application%5Bcallback_url%5D=https%3A%2F%2Foauth-openshift.apps.rh-rosa-test-cluster.z7v0.s1.devshift.org%2Foauth2callback%2Fgithub-
     1&oath_application%5Bname%5D=github
   - Click on ‘Register application’
   ```

   c. Follow the URL from the output. This creates a new OAuth application in the GitHub organization you specified.

   d. Click **Register application** to access your client ID and client secret.
e. Use the information from the GitHub application you created and continue the prompts. Enter the following values:

- Client ID: `<my_github_client_id>`
- Client Secret: `[? for help]` `<my_github_client_secret>`
- Hostname: (optional, you can leave it blank for now)
- Mapping method: `claim`

**Continued example output**

```plaintext
...?
? Client ID: `<my_github_client_id>`
? Hostname:
? Mapping method: claim
I: Configuring IDP for cluster 'rh_rosa_test_cluster'
I: Identity Provider 'github-1' has been created. You need to ensure that there is a list of cluster administrators defined. See 'rosa create user --help' for more information. To login into the console, open https://console-openshift-console.apps.rh-test-org.z7v0.s1.devshift.org and click on github-1
```

The IDP can take 1-2 minutes to be configured within your cluster.

f. Enter the following command to verify that your IDP has been configured correctly:

```bash
$ rosa list idps --cluster=<cluster_name>
```

**Example output**

```
NAME        TYPE      AUTH URL
github-1    GitHub    https://oauth-openshift.apps.rh-rosa-test-cluster1.j9n4.s1.devshift.org/oauth2callback/github-1
```

2. Log in to your cluster.

a. Enter the following command to get the **Console URL** of your cluster:

```bash
$ rosa describe cluster --cluster=<cluster_name>
```

**Example output**

```
Name:        rh-rosa-test-cluster1
ID:          1de87g7c30g75qechgh7l5b2bha6r04e
External ID: 34322be7-b2a7-45c2-af39-2c684ce624e1
API URL:     https://api.rh-rosa-test-cluster1.j9n4.s1.devshift.org:6443
Console URL: https://console-openshift-console.apps.rh-rosa-test-cluster1.j9n4.s1.devshift.org
Nodes:       Master: 3, Infra: 3, Compute: 4
Region:      us-east-2
State:       ready
Created:     May 27, 2020
```
b. Navigate to the **Console URL**, and log in using your Github credentials.

c. In the top right of the OpenShift console, click your name and click **Copy Login Command**.

d. Select the name of the IDP you added (in our case **github-1**), and click **Display Token**.

e. Copy and paste the **oc** login command into your terminal.

```
$ oc login --token=z3sgOGVDk0k4vbqo_wFqBQQTnT-nA-nQLb8XEmWnw4X --server=https://api.rh-rosa-test-cluster1.j9n4.s1.devshift.org:6443
```

Example output

```
Logged into "https://api.rh-rosa-cluster1.j9n4.s1.devshift.org:6443" as "rh-rosa-test-user" using the token provided.

You have access to 67 projects, the list has been suppressed. You can list all projects with 'oc projects'

Using project "default".
```

f. Enter a simple **oc** command to verify everything is setup properly and that you are logged in.

```
$ oc version
```

Example output

```
Client Version: 4.4.0-202005231254-4a4cd75
Server Version: 4.3.18
Kubernetes Version: v1.16.2
```

### 8.3. GRANTING CLUSTER-ADMIN ACCESS

As the user who created the cluster, add the **cluster-admin** user role to your account to have the maximum administrator privileges. These privileges are not automatically assigned to your user account when you create the cluster.

Additionally, only the user who created the cluster can grant cluster access to other **cluster-admin** or **dedicated-admin** users. Users with **dedicated-admin** access have fewer privileges. As a best practice, limit the number of **cluster-admin** users to as few as possible.

**Prerequisites**

- You have added an identity provider (IDP) to your cluster.
- You have the IDP user name for the user you are creating.
- You are logged in to the cluster.

**Procedure**

1. Give your user **cluster-admin** privileges:
Verify your user is listed as a cluster administrator:

```bash
$ rosa grant user cluster-admin --user=<idp_user_name> --cluster=<cluster_name>
```

2. Enter the following command to verify that your user now has cluster-admin access:

```bash
$ rosa list users --cluster=<cluster_name>
```

Example output

<table>
<thead>
<tr>
<th>GROUP</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>cluster-admins</td>
<td>rh-rosa-test-user</td>
</tr>
<tr>
<td>dedicated-admins</td>
<td>rh-rosa-test-user</td>
</tr>
</tbody>
</table>

3. Enter the following command to verify that your user now has cluster-admin access. A cluster administrator can run this command without errors, but a dedicated administrator cannot.

```bash
$ oc get all -n openshift-apiserver
```

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>pod/apiserver-6ndg2</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>17h</td>
</tr>
<tr>
<td>pod/apiserver-lrmxs</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>17h</td>
</tr>
<tr>
<td>pod/apiserver-tsqhz</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>17h</td>
</tr>
</tbody>
</table>

8.4. GRANTING DEDICATED-ADMIN ACCESS

Only the user who created the cluster can grant cluster access to other cluster-admin or dedicated-admin users. Users with dedicated-admin access have fewer privileges. As a best practice, grant dedicated-admin access to most of your administrators.

Prerequisites

- You have added an identity provider (IDP) to your cluster.
- You have the IDP user name for the user you are creating.
- You are logged in to the cluster.

Procedure

1. Enter the following command to promote your user to a dedicated-admin:

```bash
$ rosa grant user dedicated-admin --user=<idp_user_name> --cluster=<cluster_name>
```

2. Enter the following command to verify that your user now has dedicated-admin access:

```bash
$ rosa grant user dedicated-admin --user=<idp_user_name> --cluster=<cluster_name>
```
$ oc get groups dedicated-admins

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>USERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>dedicated-admins</td>
<td>rh-rosa-test-user</td>
</tr>
</tbody>
</table>

NOTE

A Forbidden error displays if user without dedicated-admin privileges runs this command.
CHAPTER 9. DELETING ACCESS TO A ROSA CLUSTER

Delete access to a Red Hat OpenShift Service on AWS (ROSA) cluster using the `rosa` command-line.

9.1. REVOKING DEDICATED-ADMIN ACCESS

Only the user who created the cluster can revoke access for a `dedicated-admin` users.

**Prerequisites**

- You have added an Identity Provider (IDP) to your cluster.
- You have the IDP user name for the user whose privileges you are revoking.
- You are logged in to the cluster.

**Procedure**

1. Enter the following command to revoke access for a `dedicated-admin`:
   
   ```bash
   $ rosa revoke user dedicated-admin --user=<idp_user_name> --cluster=<cluster_name>
   ```

2. Enter the following command to verify that your user no longer has `dedicated-admin` access. The user will not be listed in the output.
   
   ```bash
   $ oc get groups dedicated-admins
   ```

   **NOTE**

   A `Forbidden` error displays if user without `dedicated-admin` privileges runs this command.

9.2. REVOKING CLUSTER-ADMIN ACCESS

Only the user who created the cluster can revoke access for `cluster-admin` users.

**Prerequisites**

- You have added an Identity Provider (IDP) to your cluster.
- You have the IDP user name for the user whose privileges you are revoking.
- You are logged in to the cluster.

**Procedure**

1. Revoke the user `cluster-admin` privileges:
   
   ```bash
   $ rosa revoke user --cluster=<cluster_name> --cluster-admins=<idp_user_name>
   ```

2. Verify your user is no longer listed as a `cluster-admin`:
$ rosa list users --cluster=<cluster_name>
CHAPTER 10. DELETING A ROSA CLUSTER

Delete a Red Hat OpenShift Service on AWS (ROSA) cluster using the `rosa` command-line.

10.1. DELETING A CLUSTER

You can delete an Red Hat OpenShift Service on AWS cluster using the `rosa` CLI.

If add-ons are installed, the deletion takes longer because add-ons are uninstalled before the cluster is deleted. The amount of time depends on the number and size of the add-ons.

Procedure

1. Enter the following command to delete a cluster and watch the logs, replacing `<cluster_name>` with the name or ID of your cluster:

   ```bash
   $ rosa delete cluster --cluster=<cluster_name> --watch
   ```

2. To clean up your CloudFormation stack, enter the following command:

   ```bash
   $ rosa init --delete-stack
   ```