Red Hat OpenShift Service on AWS 4

Getting started

Setting up clusters and accounts

Last Updated: 2022-08-16

Red Hat OpenShift Service on AWS 4 Getting started

Setting up clusters and accounts

Legal Notice

Abstract

This document provides information on how to get started with Red Hat OpenShift Service on AWS (ROSA) clusters.

Table of Contents

CHAPTER 1. UNDERSTANDING THE ROSA WITH STS DEPLOYMENT WORKFLOW

Before you create a Red Hat OpenShift Service on AWS (ROSA) cluster, you must complete the AWS prerequisites, verify that the required AWS service quotas are available, and set up your environment.

This document provides an overview of the ROSA with STS deployment workflow stages and refers to detailed resources for each stage.

1.1. OVERVIEW OF THE ROSA WITH STS DEPLOYMENT WORKFLOW

The AWS Security Token Service (STS) is a global web service that provides short-term credentials for IAM or federated users. You can use AWS STS with Red Hat OpenShift Service on AWS (ROSA) to allocate temporary, limited-privilege credentials for component-specific IAM roles. The service enables cluster components to make AWS API calls using secure cloud resource management practices.

You can follow the workflow stages outlined in this section to set up and access a ROSA cluster that uses STS.

1. Complete the AWS prerequisites for ROSA with STS. To deploy a ROSA cluster with STS, your AWS account must meet the prerequisite requirements.

2. Review the required AWS service quotas. To prepare for your cluster deployment, review the AWS service quotas that are required to run a ROSA cluster.

3. Set up the environment and install ROSA using STS. Before you create a ROSA with STS cluster, you must enable ROSA in your AWS account, install and configure the required CLI tools, and verify the configuration of the CLI tools. You must also verify that the AWS Elastic Load Balancing (ELB) service role exists and that the required AWS resource quotas are available.

4. Create a ROSA cluster with STS quickly or create a cluster using customizations. Use the ROSA CLI (**rosa**) or Red Hat OpenShift Cluster Manager to create a cluster with STS. You can create a cluster quickly by using the default options, or you can apply customizations to suit the needs of your organization.

5. Access your cluster. You can configure an identity provider and grant cluster administrator privileges to the identity provider users as required. You can also access a newly-deployed cluster quickly by configuring a **cluster-admin** user.

6. Revoke access to a ROSA cluster for a user. You can revoke access to a ROSA with STS cluster from a user by using the ROSA CLI or the web console.

7. Delete a ROSA cluster. You can delete a ROSA with STS cluster by using the ROSA CLI (**rosa**). After deleting a cluster, you can delete the STS resources by using the AWS Identity and Access Management (IAM) Console.

1.2. ADDITIONAL RESOURCES

- For information about using the ROSA deployment workflow to create a cluster that does not use AWS STS, see Understanding the ROSA deployment workflow.

CHAPTER 2. GETTING STARTED WITH RED HAT OPENSHIFT SERVICE ON AWS

Follow this getting started document to quickly create a Red Hat OpenShift Service on AWS (ROSA) cluster, grant user access, deploy your first application, and learn how to revoke user access and delete your cluster.

You can create a ROSA cluster either with or without the AWS Security Token Service (STS). The procedures in this document enable you to create a cluster that uses AWS STS. For more information about using AWS STS with ROSA clusters, see Using the AWS Security Token Service .

2.1. PREREQUISITES

- You reviewed the introduction to Red Hat OpenShift Service on AWS (ROSA) , and the documentation on ROSA architecture models and architecture concepts.
- You have read the documentation on limits and scalability and the guidelines for planning your environment.
- You have reviewed the detailed AWS prerequisites for ROSA with STS.
- You have the AWS service quotas that are required to run a ROSA cluster .

2.2. SETTING UP THE ENVIRONMENT

Before you create a Red Hat OpenShift Service on AWS (ROSA) cluster, you must set up your environment by completing the following tasks:

- Enable ROSA in your AWS account
- Install and configure the required CLI tools
- Verify the configuration of the CLI tools
- Verify that the AWS Elastic Load Balancing (ELB) service role exists
- Verify that the required AWS resource quotas are available

You can follow the procedures in this section to complete these setup requirements.

2.2.1. Enabling ROSA in your AWS account

Use the steps in this procedure to enable Red Hat OpenShift Service on AWS (ROSA) in your AWS account.

Prerequisites

- You created an AWS account.

NOTE

Consider using 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 .

Procedure

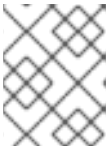
1. Sign in to the AWS Management Console.
2. Enable ROSA in your AWS account by navigating to the ROSA service and selecting **Enable OpenShift**.

2.2.2. Installing and configuring the required CLI tools

Use the following steps to install and configure the AWS, Red Hat OpenShift Service on AWS (ROSA) and OpenShift CLI tools on your workstation.

Prerequisites

- You have an AWS account.
- You created a Red Hat account.

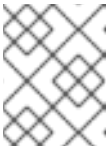


NOTE

You can create a Red Hat account by navigating to console.redhat.com and selecting **Register for a Red Hat account**

Procedure

1. Install and configure the latest AWS CLI (**aws**).
   a. Follow the AWS Command Line Interface documentation to install and configure the AWS CLI for your operating system.
   Specify your **aws_access_key_id**, **aws_secret_access_key**, and **region** in the **.aws/credentials** file. See AWS Configuration basics in the AWS documentation.



NOTE

You can alternatively use the **AWS_DEFAULT_REGION** environment variable to set the default AWS region.

   b. Query the AWS API to verify if the AWS CLI is installed and configured correctly:

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

Example output

```
<aws_account_id>    arn:aws:iam::<aws_account_id>:user/<username>  <aws_user_id>
```

2. Install and configure the latest ROSA CLI (**rosa**).
   a. Download the latest version of the **rosa** CLI for your operating system from the **Downloads** page on the OpenShift Cluster Manager.
   b. Extract the **rosa** binary file from the downloaded archive. The following example extracts the binary from a Linux tar archive:

```
$ tar xvf rosa-linux.tar.gz
```

c. Add **rosa** to your path. In the following example, the **/usr/local/bin** directory is included in the path of the user:

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

d. Verify if the **rosa** CLI tool is installed correctly by querying the **rosa** version:

```
$ rosa version
```

Example output

```
1.1.7
```

e. Optional: Generate the command completion scripts for the **rosa** CLI. The following example generates the Bash completion scripts for a Linux machine:

```
$ rosa completion bash | sudo tee /etc/bash_completion.d/rosa
```

f. Optional: Enable **rosa** command completion from your existing terminal. The following example enables Bash completion for **rosa** in an existing terminal on a Linux machine:

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

g. Log in to your Red Hat account by using the **rosa** CLI:

```
$ rosa login
```

Example output

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

Go to the URL listed in the command output to obtain an offline access token. Specify the token at the CLI prompt to log in.

NOTE

You can subsequently specify the offline access token by using the **--token="<offline_access_token>"** argument when you run the **rosa login** command.

h. Verify if you are logged in successfully and check your credentials:

```
$ rosa whoami
```

Example output

```
AWS Account ID:               <aws_account_number>
AWS Default Region:           us-east-1
```

```
AWS ARN:                      arn:aws:iam::<aws_account_number>:user/<aws_user_name>
OCM API:                      https://api.openshift.com
OCM Account ID:               <red_hat_account_id>
OCM Account Name:             Your Name
OCM Account Username:         you@domain.com
OCM Account Email:            you@domain.com
OCM Organization ID:          <org_id>
OCM Organization Name:        Your organisation
OCM Organization External ID: <external_org_id>
```

Check that the information in the output is correct before proceeding.

3. Install and configure the latest OpenShift CLI (**oc**).

   a. Use the **rosa** CLI to download the latest version of the **oc** CLI:

   ```
   $ rosa download openshift-client
   ```

   b. Extract the **oc** binary file from the downloaded archive. The following example extracts the files from a Linux tar archive:

   ```
   $ tar xvf openshift-client-linux.tar.gz
   ```

   c. Add the **oc** binary to your path. In the following example, the **/usr/local/bin** directory is included in the path of the user:

   ```
   $ sudo mv oc /usr/local/bin/oc
   ```

   d. Verify if the **oc** CLI is installed correctly:

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

   Example output

   ```
   I: Verifying whether OpenShift command-line tool is available...
   I: Current OpenShift Client Version: 4.9.12
   ```

2.2.3. Creating the ELB service role

Check if the **AWSServiceRoleForElasticLoadBalancing** AWS Elastic Load Balancing (ELB) service role exists and if not, create it.

NOTE

Error creating network Load Balancer: AccessDenied: is produced if you attempt to create a Red Hat OpenShift Service on AWS (ROSA) cluster without the AWS ELB service role in place.

Prerequisites

- You have an AWS account.
- You installed and configured the latest AWS CLI (**aws**) on your workstation.

Procedure

1. Check if the **AWSServiceRoleForElasticLoadBalancing** role exists for your AWS account:

```
$ aws iam get-role --role-name "AWSServiceRoleForElasticLoadBalancing"
```

Example output

The following example output confirms that the role exists:

```
ROLE    arn:aws:iam::<aws_account_number>:role/aws-service-
role/elasticloadbalancing.amazonaws.com/AWSServiceRoleForElasticLoadBalancing  2018-
09-27T19:49:23+00:00        Allows ELB to call AWS services on your behalf. 3600       /aws-
service-role/elasticloadbalancing.amazonaws.com/   <role_id>
AWSServiceRoleForElasticLoadBalancing
ASSUMEROLEPOLICYDOCUMENT        2012-10-17
STATEMENT       sts:AssumeRole  Allow
PRINCIPAL       elasticloadbalancing.amazonaws.com
ROLELASTUSED    2022-01-06T09:27:57+00:00       us-east-1
```

2. If the AWS ELB service role does not exist, create it:

```
$ aws iam create-service-linked-role --aws-service-name
"elasticloadbalancing.amazonaws.com"
```

2.2.4. Verifying AWS quota availability

Verify that the required resource quotas are available for your account in the default AWS region.

Prerequisites

- You have an AWS account.
- 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.

Procedure

1. Verify if the required resource quotas are available in your default region:

```
$ rosa verify quota
```

Example output

```
I: Validating AWS quota...
I: AWS quota ok. If cluster installation fails, validate actual AWS resource usage against
https://docs.openshift.com/rosa/rosa_getting_started/rosa-required-aws-service-quotas.html
```

2.3. CREATING A ROSA CLUSTER WITH STS

Choose from one of the following methods to deploy a Red Hat OpenShift Service on AWS (ROSA) cluster that uses the AWS Security Token Service (STS). In both scenarios, you can deploy your cluster by using Red Hat OpenShift Cluster Manager or the ROSA CLI (**rosa**):

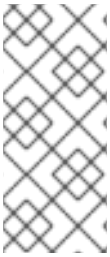
- **Creating a ROSA cluster with STS using the default options** You can create a ROSA cluster with STS quickly by using the default options and automatic STS resource creation.
- **Creating a ROSA cluster with STS using customizations** You can create a ROSA cluster with STS using customizations. You can also choose between the **auto** and **manual** modes when creating the required STS resources.

Additional resources

- For detailed steps to deploy a ROSA cluster without STS, see Creating a ROSA cluster without AWS STS and Creating an AWS PrivateLink cluster on ROSA .
- For information about the account-wide IAM roles and policies that are required for ROSA deployments that use STS, see Account-wide IAM role and policy reference .
- For details about using the **auto** and **manual** modes to create the required STS resources, see Understanding the auto and manual deployment modes .
- For information about the update life cycle for ROSA, see Red Hat OpenShift Service on AWS update life cycle.

2.4. CREATING A CLUSTER ADMINISTRATOR USER FOR QUICK CLUSTER ACCESS

Before configuring an identity provider, you can create a user with **cluster-admin** privileges for immediate access to your Red Hat OpenShift Service on AWS (ROSA) cluster.



NOTE

The cluster administrator user is useful when you need quick access to a newly deployed cluster. However, consider configuring an identity provider and granting cluster administrator privileges to the identity provider users as required. For more information about setting up an identity provider for your ROSA cluster, see *Configuring an identity provider and granting cluster access.*

Prerequisites

- You have an AWS account.
- 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

1. Create a cluster administrator user:

```
$ rosa create admin --cluster=<cluster_name> 1
```

```
-
```

1 Replace **<cluster_name>** with the name of your cluster.

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>'.
I: Please securely store this generated password. If you lose this password you can delete and recreate the cluster admin user.
I: To login, run the following command:

   oc login https://api.example-cluster.wxyz.p1.openshiftapps.com:6443 --username cluster-admin --password d7Rca-Ba4jy-YeXhs-WU42J

I: It may take up to a minute for the account to become active.
```

NOTE

It might take approximately one minute for the **cluster-admin** user to become active.

2. Log in to the cluster through the CLI:

   a. Run the command provided in the output of the preceding step to log in:

```
$ oc login <api_url> --username cluster-admin --password <cluster_admin_password> 1
```

1 Replace **<api_url>** and **<cluster_admin_password>** with the API URL and cluster administrator password for your environment.

   b. Verify if you are logged in to the ROSA cluster as the **cluster-admin** user:

```
$ oc whoami
```

Example output

```
cluster-admin
```

Additional resource

- For steps to log in to the ROSA web console, see Accessing a cluster through the web console

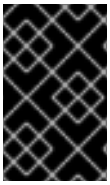
2.5. CONFIGURING AN IDENTITY PROVIDER AND GRANTING CLUSTER ACCESS

Red Hat OpenShift Service on AWS (ROSA) includes a built-in OAuth server. After your ROSA cluster is created, you must configure OAuth to use an identity provider. You can then add members to your configured identity provider to grant them access to your cluster.

You can also grant the identity provider users with **cluster-admin** or **dedicated-admin** privileges as required.

2.5.1. Configuring an identity provider

You can configure different identity provider types for your Red Hat OpenShift Service on AWS (ROSA) cluster. Supported types include GitHub, GitHub Enterprise, GitLab, Google, LDAP, OpenID Connect and HTPasswd identity providers.



IMPORTANT

The HTPasswd identity provider option is included only to enable the creation of a single, static administration user. HTPasswd is not supported as a general-use identity provider for Red Hat OpenShift Service on AWS.

The following procedure configures a GitHub identity provider as an example.

Prerequisites

- You have an AWS account.
- 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.
- You have a GitHub user account.

Procedure

1. Go to github.com and log in to your GitHub account.
2. If you do not have an existing GitHub organization to use for identity provisioning for your ROSA cluster, create one. Follow the steps in the GitHub documentation.
3. Configure a GitHub identity provider for your cluster that is restricted to the members of your GitHub organization.
   a. Configure an identity provider using the interactive mode:

```
$ rosa create idp --cluster=<cluster_name> --interactive (1)
```

(1) Replace **<cluster_name>** with the name of your cluster.

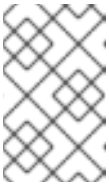
Example output

```
I: Interactive mode enabled.
Any optional fields can be left empty and a default will be selected.
? Type of identity provider: github
? Identity provider name: github-1
? Restrict to members of: organizations
```

```
? GitHub organizations: <github_org_name> (1)
? To use GitHub as an identity provider, you must first register the application:
  - Open the following URL:
    https://github.com/organizations/<github_org_name>/settings/applications/new?
oauth_application%5Bcallback_url%5D=https%3A%2F%2Foauth-openshift.apps.
<cluster_name>/<random_string>.p1.openshiftapps.com%2Foauth2callback%2Fgithub-
1&oauth_application%5Bname%5D=
<cluster_name>&oauth_application%5Burl%5D=https%3A%2F%2Fconsole-openshift-
console.apps.<cluster_name>/<random_string>.p1.openshiftapps.com
  - Click on 'Register application'
...
```

(1) Replace **<github_org_name>** with the name of your GitHub organization.

b. Follow the URL in the output and select **Register application** to register a new OAuth application in your GitHub organization. By registering the application, you enable the OAuth server that is built into ROSA to authenticate members of your GitHub organization into your cluster.



NOTE

The fields in the **Register a new OAuth application** GitHub form are automatically filled with the required values through the URL defined by the **rosa** CLI tool.

c. Use the information from your GitHub OAuth application page to populate the remaining **rosa create idp** interactive prompts.

Continued example output

```
...
? Client ID: <github_client_id> (1)
? Client Secret: [? for help] <github_client_secret> (2)
? GitHub Enterprise Hostname (optional):
? Mapping method: claim (3)
I: Configuring IDP for cluster '<cluster_name>'
I: Identity Provider 'github-1' has been created.
  It will take up to 1 minute for this configuration to be enabled.
  To add cluster administrators, see 'rosa grant user --help'.
  To login into the console, open https://console-openshift-console.apps.<cluster_name>.
<random_string>.p1.openshiftapps.com and click on github-1.
```

(1) Replace **<github_client_id>** with the client ID for your GitHub OAuth application.

(2) Replace **<github_client_secret>** with a client secret for your GitHub OAuth application.

(3) Specify **claim** as the mapping method.

NOTE

It might take approximately two minutes for the identity provider configuration to become active. If you have configured a **cluster-admin** user, you can watch the OAuth pods redeploy with the updated configuration by running **oc get pods -n openshift-authentication --watch**.

d. Enter the following command to verify that the identity provider has been configured correctly:

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

Example output

```
NAME       TYPE      AUTH URL
github-1   GitHub    https://oauth-openshift.apps.<cluster_name>.
<random_string>.p1.openshiftapps.com/oauth2callback/github-1
```

Additional resource

- For detailed steps to configure each of the supported identity provider types, see Configuring identity providers for STS

2.5.2. Granting user access to a cluster

You can grant a user access to your Red Hat OpenShift Service on AWS (ROSA) cluster by adding them to your configured identity provider.

You can configure different types of identity providers for your ROSA cluster. The following example procedure adds a user to a GitHub organization that is configured for identity provision to the cluster.

Prerequisites

- You have an AWS account.
- 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.
- You have a GitHub user account.
- You have configured a GitHub identity provider for your cluster.

Procedure

1. Navigate to github.com and log in to your GitHub account.
2. Invite users that require access to the ROSA cluster to your GitHub organization. Follow the steps in Inviting users to join your organization in the GitHub documentation.

2.5.3. Granting administrator privileges to a user

After you have added a user to your configured identity provider, you can grant the user **cluster-admin** or **dedicated-admin** privileges for your Red Hat OpenShift Service on AWS (ROSA) cluster.

Prerequisites

- You have an AWS account.
- 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.
- You have configured a GitHub identity provider for your cluster and added identity provider users.

Procedure

- To configure **cluster-admin** privileges for an identity provider user:
  a. Grant the user **cluster-admin** privileges:

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

(1)

(1) Replace **<idp_user_name>** and **<cluster_name>** with the name of the identity provider user and your cluster name.

Example output

```
I: Granted role 'cluster-admins' to user '<idp_user_name>' on cluster '<cluster_name>'
```

  b. Verify if the user is listed as a member of the **cluster-admins** group:

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

Example output

```
ID                GROUPS
<idp_user_name>   cluster-admins
```

- To configure **dedicated-admin** privileges for an identity provider user:
  a. Grant the user **dedicated-admin** privileges:

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

Example output

```
I: Granted role 'dedicated-admins' to user '<idp_user_name>' on cluster '<cluster_name>'
```

  b. Verify if the user is listed as a member of the **dedicated-admins** group:

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

Example output

```
ID              GROUPS
<idp_user_name>   dedicated-admins
```

2.6. ACCESSING A CLUSTER THROUGH THE WEB CONSOLE

After you have created a cluster administrator user or added a user to your configured identity provider, you can log into your Red Hat OpenShift Service on AWS (ROSA) cluster through the web console.

Prerequisites

- You have an AWS account.
- 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.
- You have created a cluster administrator user or added your user account to the configured identity provider.

Procedure

1. Obtain the console URL for your cluster:

```
$ rosa describe cluster -c <cluster_name> | grep Console
```
(1)

(1) Replace **<cluster_name>** with the name of your cluster.

Example output

```
Console URL:                https://console-openshift-console.apps.example-
cluster.wxyz.p1.openshiftapps.com
```

2. Go to the console URL in the output of the preceding step and log in.
   - If you created a **cluster-admin** user, log in by using the provided credentials.
   - If you configured an identity provider for your cluster, select the identity provider name in the **Log in with...** dialog and complete any authorization requests that are presented by your provider.

2.7. DEPLOYING AN APPLICATION FROM THE DEVELOPER CATALOG

From the Red Hat OpenShift Service on AWS web console, you can deploy a test application from the Developer Catalog and expose it with a route.

Prerequisites

- You logged in to OpenShift Cluster Manager.
- You created an Red Hat OpenShift Service on AWS cluster.
- You configured an identity provider for your cluster.
- You added your user account to the configured identity provider.

Procedure

1. From OpenShift Cluster Manager, click **Open console**.
2. In the **Administrator** perspective, select **Home → Projects → Create Project**.
3. Enter a name for your project and optionally add a **Display Name** and **Description**.
4. Click **Create** to create the project.
5. Switch to the **Developer** perspective and select **+Add**. Make sure that the selected **Project** is the one that you just created.
6. In the **Developer Catalog** dialog, select **All services**.
7. In the **Developer Catalog** page, select **Languages → JavaScript** from the menu.
8. Click **Node.js**, and then click **Create Application** to open the **Create Source-to-Image Application** page.

NOTE

You might need to click **Clear All Filters** to display the **Node.js** option.

9. In the **Git** section, click **Try Sample**.
10. Add a unique name in the **Name** field. The value will be used to name the associated resources.
11. Confirm that **Deployment** and **Create a route to the application** are selected.
12. Click **Create** to deploy the application. It will take a few minutes for the pods to deploy.
13. Optional: Check the status of the pods in the **Topology** pane by selecting your **nodejs** app and reviewing its sidebar. You must wait for the **nodejs** build to complete and for the **nodejs** pod to be in a **Running** state before continuing.
14. When the deployment is complete, click route URL for the application, which has a format similar to the following:

```
http://nodejs-<project>.<cluster_name>.<hash>.<region>.openshiftapps.com/
```

A new tab in your browser opens with a message similar to the following.

```
Welcome to your Node.js application on OpenShift
```

15. Optional: Delete the application and clean up the resources that you created:

a. In the **Administrator** perspective, navigate to **Home → Projects**.

b. Click the action menu for your project and select **Delete Project**.

2.8. REVOKING ADMINISTRATOR PRIVILEGES AND USER ACCESS

You can revoke **cluster-admin** or **dedicated-admin** privileges from a user by using the ROSA CLI (**rosa**).

To revoke cluster access from a user, you must remove the user from your configured identity provider.

Follow the procedures in this section to revoke administrator privileges or cluster access from a user.

2.8.1. Revoking administrator privileges from a user

Follow the steps in this section to revoke **cluster-admin** or **dedicated-admin** privileges from a user.

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.
- You have configured a GitHub identity provider for your cluster and added an identity provider user.
- You granted **cluster-admin** or **dedicated-admin** privileges to a user.

Procedure

- To revoke **cluster-admin** privileges from an identity provider user:

  a. Revoke the **cluster-admin** privilege:

  ```
  $ rosa revoke user cluster-admin --user=<idp_user_name> --cluster=<cluster_name>
  ```
  (1)

  (1) Replace **<idp_user_name>** and **<cluster_name>** with the name of the identity provider user and your cluster name.

  Example output

  ```
  ? Are you sure you want to revoke role cluster-admins from user <idp_user_name> in
  cluster <cluster_name>? Yes
  I: Revoked role 'cluster-admins' from user '<idp_user_name>' on cluster '<cluster_name>'
  ```

  b. Verify that the user is not listed as a member of the **cluster-admins** group:

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

Example output

```
W: There are no users configured for cluster '<cluster_name>'
```

- To revoke **dedicated-admin** privileges from an identity provider user:

  a. Revoke the **dedicated-admin** privilege:

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

  Example output

  ```
  ? Are you sure you want to revoke role dedicated-admins from user <idp_user_name> in cluster <cluster_name>? Yes
  I: Revoked role 'dedicated-admins' from user '<idp_user_name>' on cluster '<cluster_name>'
  ```

  b. Verify that the user is not listed as a member of the **dedicated-admins** group:

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

  Example output

  ```
  W: There are no users configured for cluster '<cluster_name>'
  ```

2.8.2. Revoking user access to a cluster

You can revoke cluster access for an identity provider user by removing them from your configured identity provider.

You can configure different types of identity providers for your ROSA cluster. The following example procedure revokes cluster access for a member of a GitHub organization that is configured for identity provision to the cluster.

Prerequisites

- You have a ROSA cluster.
- You have a GitHub user account.
- You have configured a GitHub identity provider for your cluster and added an identity provider user.

Procedure

1. Navigate to github.com and log in to your GitHub account.
2. Remove the user from your GitHub organization. Follow the steps in Removing a member from your organization in the GitHub documentation.

2.9. DELETING A ROSA CLUSTER AND THE AWS STS RESOURCES

You can delete a ROSA cluster that uses the AWS Security Token Service (STS) by using the ROSA CLI (**rosa**). You can also use the ROSA CLI to delete the AWS Identity and Access Management (IAM) account-wide roles, the cluster-specific Operator roles, and the OpenID Connect (OIDC) provider. To delete the account-wide inline and Operator policies, you can use the AWS IAM Console.

IMPORTANT

Account-wide IAM roles and policies might be used by other ROSA clusters in the same AWS account. You must only remove the resources if they are not required by other clusters.

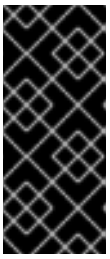
Prerequisites

- 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

1. Delete a cluster and watch the logs, replacing **<cluster_name>** with the name or ID of your cluster:

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



IMPORTANT

You must wait for the cluster deletion to complete before you remove the IAM roles, policies, and OIDC provider. The account-wide roles are required to delete the resources created by the installer. The cluster-specific Operator roles are required to clean-up the resources created by the OpenShift Operators. The Operators use the OIDC provider to authenticate.

2. Delete the OIDC provider that the cluster Operators use to authenticate:

```
$ rosa delete oidc-provider -c <cluster_id> --mode auto (1)
```

(1) Replace **<cluster_id>** with the ID of the cluster.

NOTE

You can use the **-y** option to automatically answer yes to the prompts.

3. Delete the cluster-specific Operator IAM roles:

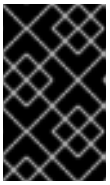
```
$ rosa delete operator-roles -c <cluster_id> --mode auto (1)
```

(1) Replace **<cluster_id>** with the ID of the cluster.

4. Delete the account-wide roles:

```
$ rosa delete account-roles --prefix <prefix> --mode auto 1
```

1 You must include the **--<prefix>** argument. Replace **<prefix>** with the prefix of the account-wide roles to delete. If you did not specify a custom prefix when you created the account-wide roles, specify the default prefix, **ManagedOpenShift**.



IMPORTANT

Account-wide IAM roles and policies might be used by other ROSA clusters in the same AWS account. You must only remove the resources if they are not required by other clusters.

5. Delete the account-wide inline and Operator IAM policies that you created for ROSA deployments that use STS:

   a. Log in to the AWS IAM Console.

   b. Navigate to **Access management → Policies** and select the checkbox for one of the account-wide policies.

   c. With the policy selected, click on **Actions → Delete** to open the delete policy dialog.

   d. Enter the policy name to confirm the deletion and select **Delete** to delete the policy.

   e. Repeat this step to delete each of the account-wide inline and Operator policies for the cluster.

2.10. NEXT STEPS

- Adding services to a cluster using the OpenShift Cluster Manager console
- Managing compute nodes
- Configuring the monitoring stack
- Installing logging add-on services

2.11. ADDITIONAL RESOURCES

- For more information about setting up accounts and ROSA clusters using AWS STS, see Understanding the ROSA with STS deployment workflow
- For information about setting up accounts and ROSA clusters without using AWS STS, see Understanding the ROSA deployment workflow
- For documentation on upgrading your cluster, see Upgrading ROSA clusters