Red Hat OpenShift Service on AWS 4

Getting started

Setting up clusters and accounts
Setting up clusters and accounts
Abstract
This document provides information on how to get started with Red Hat OpenShift Service on AWS (ROSA) clusters.
Table of Contents

CHAPTER 1. GETTING STARTED WITH RED HAT OPENSOURCE SERVICE ON AWS .......................... 5
  1.1. PREREQUISITES .......................... 5
  1.2. SETTING UP THE ENVIRONMENT .................. 5
    1.2.1. Enabling ROSA in your AWS account .......... 5
    1.2.2. Installing and configuring the required CLI tools 6
    1.2.3. Creating the ELB service role ............... 8
    1.2.4. Verifying AWS quota availability .......... 9
  1.3. CREATING A ROSA CLUSTER WITH STS USING THE DEFAULT OPTIONS ......... 9
  1.4. CREATING A CLUSTER ADMINISTRATOR USER FOR QUICK CLUSTER ACCESS ....... 11
  1.5. CONFIGURING AN IDENTITY PROVIDER AND GRANTING CLUSTER ACCESS .......... 13
    1.5.1. Configuring an identity provider .......... 13
    1.5.2. Granting user access to a cluster .......... 15
    1.5.3. Granting administrator privileges to a user ... 15
  1.6. ACCESSING A CLUSTER THROUGH THE WEB CONSOLE .................. 17
  1.7. DEPLOYING AN APPLICATION FROM THE DEVELOPER CATALOG .............. 17
  1.8. REVOKING ADMINISTRATOR PRIVILEGES AND USER ACCESS .................. 19
    1.8.1. Revoking administrator privileges from a user .... 19
    1.8.2. Revoking user access to a cluster .......... 20
  1.9. DELETING A ROSA CLUSTER AND THE AWS STS RESOURCES ............ 20
  1.10. NEXT STEPS .................................. 22
  1.11. ADDITIONAL RESOURCES ........................ 22

CHAPTER 2. REQUIRED AWS SERVICE QUOTAS ........................................... 23
  2.1. REQUIRED AWS SERVICE QUOTAS .............. 23
  2.2. NEXT STEPS .................................. 24

CHAPTER 3. SETTING UP THE ENVIRONMENT FOR USING STS ...................... 25
  3.1. SETTING UP THE ENVIRONMENT FOR STS .............. 25
  3.2. NEXT STEPS .................................. 28
  3.3. ADDITIONAL RESOURCES ........................ 28

CHAPTER 4. CREATING A ROSA CLUSTER WITH STS QUICKLY .................. 29
  4.1. CREATING A ROSA CLUSTER WITH STS USING THE DEFAULT OPTIONS ...... 29
  4.2. NEXT STEPS .................................. 30
  4.3. ADDITIONAL RESOURCES ........................ 30

CHAPTER 5. CREATING A ROSA CLUSTER WITH STS USING CUSTOMIZATIONS .... 31
  5.1. SUPPORT CONSIDERATIONS FOR ROSA CLUSTERS WITH STS ............ 31
  5.2. CREATING A CLUSTER WITH STS USING CUSTOMIZATIONS .......... 31
  5.3. NEXT STEPS .................................. 37
  5.4. ADDITIONAL RESOURCES ........................ 37

CHAPTER 6. ABOUT IAM RESOURCES FOR ROSA CLUSTERS THAT USE STS .......... 38
  6.1. ACCOUNT-WIDE IAM ROLE AND POLICY REFERENCE ............ 38
    6.1.1. Account-wide IAM role and policy AWS CLI reference ........ 52
  6.2. CLUSTER-SPECIFIC OPERATOR IAM ROLE REFERENCE .......... 54
    6.2.1. Operator IAM role AWS CLI reference ............ 55
  6.3. OIDC PROVIDER REQUIREMENTS FOR OPERATOR AUTHENTICATION .... 56
    6.3.1. OIDC provider AWS CLI reference .......... 57

CHAPTER 7. INTERACTIVE CLUSTER CREATION MODE REFERENCE ................ 58
  7.1. UNDERSTANDING THE INTERACTIVE CLUSTER CREATION MODE OPTIONS .... 58
  7.2. ADDITIONAL RESOURCES ........................ 60
CHAPTER 1. 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.

1.1. PREREQUISITES

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

1.2. 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.

1.2.1. 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.

1.2.2. 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```
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...
```

1.2.3. 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:

   ```bash
   $ 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:

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

1.2.4. 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:

   ```bash
   $ 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
   ```

1.3. CREATING A ROSA CLUSTER WITH STS USING THE DEFAULT OPTIONS
Through the Red Hat OpenShift Service on AWS CLI (rosa), you can quickly create an OpenShift cluster that uses the AWS Security Token Service (STS).

Additionally, you can use auto mode to immediately create the required AWS Identity and Access Management (IAM) resources using the current AWS account. auto mode is used in the following procedure to immediately create the account-wide IAM roles and policies, including the Operator policies, as well as the OpenID Connect (OIDC) identity provider.

Prerequisites

- You have completed the AWS prerequisites for ROSA with STS.
- You have available AWS service quotas.
- You have enabled the ROSA service in the AWS Console.
- 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 verified that the AWS Elastic Load Balancing (ELB) service role exists in your AWS account.

NOTE

To successfully install ROSA 4.10 clusters, use ROSA CLI 1.1.11 or above.

Procedure

1. Create the required account-wide roles and policies, including the Operator policies:

   $ rosa create account-roles --mode auto

   NOTE

   When using auto mode, you can optionally specify the -y argument to bypass the interactive prompts and automatically confirm operations.

2. Create a cluster with STS using the defaults. When you use the defaults, the latest stable OpenShift version is installed:

   $ rosa create cluster --cluster-name <cluster_name> --sts --mode auto

   Replace <cluster_name> with the name of your cluster.

   NOTE

   When you specify --mode auto, the rosa create cluster command creates the cluster-specific Operator IAM roles and the OIDC provider automatically. The Operators use the OIDC provider to authenticate.
3. Check the status of your cluster:

```bash
$ rosa describe cluster --cluster <cluster_name|cluster_id>
```

The following State field changes are listed in the output as the cluster installation progresses:

- **waiting** *(Waiting for OIDC configuration)*
- **pending** *(Preparing account)*
- **installing** *(DNS setup in progress)*
- **installing**
- **ready**

**NOTE**

If installation fails or the State field does not change to ready after about 40 minutes, check the installation troubleshooting documentation for more details.

4. Track the progress of the cluster creation by watching the OpenShift installer logs:

```bash
$ rosa logs install --cluster <cluster_name|cluster_id> --watch
```

Specify the --watch flag to watch for new log messages as the installation progresses. This argument is optional.

### 1.4. 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:

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

   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:

   ```bash
   ```

   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:

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

   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:

   ```bash
   $ 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
1.5. 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.

1.5.1. 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
   ```

   Replace `<cluster_name>` with the name of your cluster.

**Example output**
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**

```bash
...?

? Client ID: <github_client_id>  
? Client Secret: [?] for help <github_client_secret>  
? GitHub Enterprise Hostname (optional): 
? Mapping method: claim

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**

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>AUTH URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>github-1</td>
<td>GitHub</td>
<td><a href="https://oauth-openshift.apps">https://oauth-openshift.apps</a>.&lt;cluster_name&gt;.&lt;random_string&gt;.p1.openshiftapps.com/oauth2callback/github-1</td>
</tr>
</tbody>
</table>

Additional resource

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

1.5.2. 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.

1.5.3. 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> 
  ```

  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:
1.6. 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:

   ```bash
   $ rosa describe cluster -c <cluster_name> | grep Console
   ```

   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.

1.7. 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:
In the **Administrator** perspective, navigate to **Home → Projects**.

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

### 1.8. 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.

#### 1.8.1. 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>
     ```

     Replace `<idp_user_name>` and `<cluster_name>` with the name of the identity provider user and your 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>'

1.8.2. 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.

1.9. 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
   ```

   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
   ```

   Replace `<cluster_id>` with the ID of the cluster.
4. Delete the account-wide roles:

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

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.

1.10. 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

1.11. 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
CHAPTER 2. 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.

2.1. 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-DemandStandard (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>50</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>50</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>

2.2. NEXT STEPS

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

3.1. 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
e. 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. Source the scripts to enable `rosa` command completion from your existing terminal. The following example sources the Bash completion scripts for `rosa` on a Linux machine:

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

5. 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://console.redhat.com/openshift/token/rosa

? Copy the token and paste it here: `<myofflineaccess_token>`

**Example output continued**

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

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

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

**Example output**

I: Validating AWS quota...
I: AWS quota ok

**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.

7. 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 OpenShift Cluster Manager for now.

   $ rosa whoami

   **Example output**
   
   - 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: 1DzGldhqwEyWt8UUXQhSoWaaaaa
   - OCM Account Name: Your Name
   - OCM Account Username: you@domain.com
   - OCM Account Email: you@domain.com
   - OCM Organization ID: 1HopHfA2hcnpup5gCr2uh5aaaaa
   - OCM Organization Name: Red Hat
   - OCM Organization External ID: 000000

8. 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

**Create roles**

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

### 3.2. NEXT STEPS

- Create a ROSA cluster with STS quickly or create a cluster using customizations.

### 3.3. ADDITIONAL RESOURCES

- AWS Prerequisites
- Required AWS service quotas and increase requests
CHAPTER 4. CREATING A ROSA CLUSTER WITH STS QUICKLY

Use the Red Hat OpenShift Service on AWS CLI (rosa) with the default options to quickly create an OpenShift cluster that uses the AWS Security Token Service (STS).

4.1. CREATING A ROSA CLUSTER WITH STS USING THE DEFAULT OPTIONS

Through the Red Hat OpenShift Service on AWS CLI (rosa), you can quickly create an OpenShift cluster that uses the AWS Security Token Service (STS).

Additionally, you can use auto mode to immediately create the required AWS Identity and Access Management (IAM) resources using the current AWS account. auto mode is used in the following procedure to immediately create the account-wide IAM roles and policies, including the Operator policies, as well as the OpenID Connect (OIDC) identity provider.

Prerequisites

- You have completed the AWS prerequisites for ROSA with STS.
- You have available AWS service quotas.
- You have enabled the ROSA service in the AWS Console.
- 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 verified that the AWS Elastic Load Balancing (ELB) service role exists in your AWS account.

**NOTE**

To successfully install ROSA 4.10 clusters, use ROSA CLI 1.1.11 or above.

Procedure

1. Create the required account-wide roles and policies, including the Operator policies:

   ```
   $ rosa create account-roles --mode auto
   ```

   **NOTE**

   When using auto mode, you can optionally specify the -y argument to bypass the interactive prompts and automatically confirm operations.

2. Create a cluster with STS using the defaults. When you use the defaults, the latest stable OpenShift version is installed:

   ```
   $ rosa create cluster --cluster-name <cluster_name> --sts --mode auto
   ```

   Replace `<cluster_name>` with the name of your cluster.
NOTE
When you specify `--mode auto`, the `rosa create cluster` command creates the cluster-specific Operator IAM roles and the OIDC provider automatically. The Operators use the OIDC provider to authenticate.

3. Check the status of your cluster:

   ```bash
   $ rosa describe cluster --cluster <cluster_name|cluster_id>
   ```

   The following `State` field changes are listed in the output as the cluster installation progresses:

   - **waiting** *(Waiting for OIDC configuration)*
   - **pending** *(Preparing account)*
   - **installing** *(DNS setup in progress)*
   - **installing**
   - **ready**

   **NOTE**
   If installation fails or the `State` field does not change to `ready` after about 40 minutes, check the installation troubleshooting documentation for more details.

4. Track the progress of the cluster creation by watching the OpenShift installer logs:

   ```bash
   $ rosa logs install --cluster <cluster_name|cluster_id> --watch
   ```

   Specify the `--watch` flag to watch for new log messages as the installation progresses. This argument is optional.

4.2. NEXT STEPS

- Accessing a ROSA cluster

4.3. ADDITIONAL RESOURCES

- For more information about the AWS Identity Access Management (IAM) resources required to deploy Red Hat OpenShift Service on AWS with STS, see [About IAM resources for clusters that use STS](#).

- For information about the prerequisites to installing ROSA with STS, see [AWS prerequisites for ROSA with STS](#).

- For more information about using OpenID Connect (OIDC) identity providers in AWS IAM, see [Creating OpenID Connect (OIDC) identity providers](#) in the AWS documentation.

- For more information about troubleshooting ROSA cluster deployments, see [Troubleshooting cluster deployments](#).
CHAPTER 5. CREATING A ROSA CLUSTER WITH STS USING CUSTOMIZATIONS

Use the Red Hat OpenShift Service on AWS CLI (rosa) to create an OpenShift cluster with the AWS Security Token Service (STS) using customizations.

5.1. SUPPORT CONSIDERATIONS FOR ROSA CLUSTERS WITH STS

The supported way of creating a Red Hat OpenShift Service on AWS cluster that uses the AWS Security Token Service (STS) is by using the steps described in this product documentation.

You can use manual mode with the Red Hat OpenShift Service on AWS CLI (rosa) to generate the AWS Identity and Access Management (IAM) policy files and aws commands that are required to install the STS resources.

**IMPORTANT**

The files and aws commands are generated for review purposes only and must not be modified in any way. Red Hat cannot provide support for ROSA clusters that have been deployed by using modified versions of the policy files or aws commands.

5.2. CREATING A CLUSTER WITH STS USING CUSTOMIZATIONS

When you create a Red Hat OpenShift Service on AWS (ROSA) cluster that uses the AWS Security Token Service (STS), you can customize your installation interactively. When you run rosa create cluster --interactive at cluster creation time, you are presented with a series of interactive prompts that enable you to customize your deployment. For more information, see Interactive cluster creation mode reference.

There are two rosa CLI modes for deploying a cluster with STS:

- **manual** mode. With this mode, the rosa CLI generates the aws commands needed to create the required AWS Identity and Access Management (IAM) roles and policies, and an OpenID Connect (OIDC) provider. The corresponding policy JSON files are also saved to the current directory. This enables you to review the details before running the aws commands manually.

- **auto** mode. You can use this mode to immediately create the required AWS Identity and Access Management (IAM) resources using the current AWS account.

**IMPORTANT**

Only public and AWS PrivateLink clusters are supported with STS. Regular private clusters (non-PrivateLink) are not available for use with STS.

**NOTE**

AWS Shared VPCs are not currently supported for ROSA installations.

**Prerequisites**

- You have completed the AWS prerequisites for ROSA with STS.
- You have available AWS service quotas.
• You have enabled the ROSA service in the AWS Console.

• You have installed and configured the latest AWS, ROSA, and oc CLIs on your installation host.

• If you are using a customer-managed AWS Key Management Service (KMS) key for encryption, you have created a symmetric KMS key and you have the key ID and Amazon Resource Name (ARN). For more information about creating AWS KMS keys, see the AWS documentation.

**NOTE**

To successfully install ROSA 4.10 clusters, use ROSA CLI 1.1.11 or above.

**Procedure**

1. Create the required account-wide roles and policies, including the Operator policies:
   
   a. Generate the IAM policy JSON files in the current working directory and output the `aws` CLI commands for review:

   ```bash
   $ rosa create account-roles --mode manual
   
   manual mode generates the `aws` CLI commands and JSON files needed to create the account-wide roles and policies. After review, you must run the commands manually to create the resources.
   ```

   b. After review, run the `aws` commands manually to create the roles and policies. Alternatively, you can run the preceding command using `--mode auto` to run the `aws` commands immediately.

2. (Optional) If you are using your own AWS KMS key to encrypt the control plane data volumes and the persistent volumes (PVs) for your applications, add the ARN for the account-wide installer role to your KMS key policy.

   a. Save the key policy for your KMS key to a file on your local machine. The following example saves the output to `kms-key-policy.json` in the current working directory:

   ```bash
   $ aws kms get-key-policy --key-id <key_id_or_arn> --policy-name default --output text > kms-key-policy.json
   
   Replace `<key_id_or_arn>` with the ID or ARN of your KMS key.
   ```

   b. Add the ARN for the account-wide installer role that you created in the preceding step to the `Statement.Principal.AWS` section in the file. In the following example, the ARN for the default `ManagedOpenShift-Installer-Role` role is added:

   ```json
   {
     "Version": "2012-10-17",
     "Id": "key-default-1",
     "Statement": [
       {
         "Sid": "Enable IAM User Permissions",
         "Effect": "Allow",
         "Principal": {
   ```
c. Apply the changes to your KMS key policy:

```bash
$ aws kms put-key-policy --key-id <key_id_or_arn> \
   --policy file://kms-key-policy.json \
   --policy-name default
```

1 Replace `<key_id_or_arn>` with the ID or ARN of your KMS key.

2 You must include the `file://` prefix when referencing a key policy in a local file.

You can reference the ARN of your KMS key when you create the cluster in the next step.

3. Create a cluster with STS using custom installation options. You can use the `--interactive` mode to interactively specify custom settings:

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

Example output

```
I: Interactive mode enabled.
Any optional fields can be left empty and a default will be selected.
? Cluster name: <cluster_name>
? OpenShift version: 4.8.9

I: Using arn:aws:iam::<aws_account_id>:role/ManagedOpenShift-Installer-Role for the Installer role
I: Using arn:aws:iam::<aws_account_id>:role/ManagedOpenShift-ControlPlane-Role for the ControlPlane role
I: Using arn:aws:iam::<aws_account_id>:role/ManagedOpenShift-Worker-Role for the Worker role
I: Using arn:aws:iam::<aws_account_id>:role/ManagedOpenShift-Support-Role for the Support role
? External ID (optional):
? Operator roles prefix: <cluster_name>---<random_string>
? Multiple availability zones (optional): No
? AWS region: us-east-1
? PrivateLink cluster (optional): No
? Install into an existing VPC (optional): No
? Enable Customer Managed key (optional): No
? Compute nodes instance type (optional):
```
When creating the cluster, the listed OpenShift version options include the major, minor, and patch versions, for example 4.8.9.

If more than one matching set of account-wide roles are available in your account for a cluster version, an interactive list of options is provided.

Multiple availability zones are recommended for production workloads. The default is a single availability zone.

Enable this option if you are using your own AWS KMS key to encrypt the control plane data volumes and the PVs for your applications. Specify the ARN for the KMS key that you added the account-wide role ARN to in the preceding step.

Enable this option only if your use case requires etcd key value encryption in addition to the control plane storage encryption that encrypts the etcd volumes by default. With this option, the etcd key values are encrypted, but not the keys.

IMPORTANT

By enabling etcd encryption for the key values in etcd, you will incur a performance overhead of approximately 20%. The overhead is a result of introducing this second layer of encryption, in addition to the default control plane storage encryption that encrypts the etcd volumes. Red Hat recommends that you enable etcd encryption only if you specifically require it for your use case.

The output includes a custom command that you can run to create a cluster with the same configuration in the future.
As an alternative to using the --interactive mode, you can specify the customization options directly when you run `rosa create cluster`. Run `rosa create cluster --help` to view a list of available CLI options.

**IMPORTANT**

You must complete the following steps to create the Operator IAM roles and the OpenID Connect (OIDC) provider to move the state of the cluster to **ready**.

4. Create the cluster-specific Operator IAM roles:
   
a. Generate the Operator IAM policy JSON files in the current working directory and output the `aws` CLI commands for review:

   ```bash
   $ rosa create operator-roles --mode manual --cluster <cluster_name|cluster_id>  
   ``

   1 **manual** mode generates the `aws` CLI commands and JSON files needed to create the Operator roles. After review, you must run the commands manually to create the resources.

   b. After review, run the `aws` commands manually to create the Operator IAM roles and attach the managed Operator policies to them. Alternatively, you can run the preceding command again using **--mode auto** to run the `aws` commands immediately.

5. Create the OpenID Connect (OIDC) provider that the cluster Operators use to authenticate:

   ```bash
   $ rosa create oidc-provider --mode auto --cluster <cluster_name|cluster_id>  
   ``

   1 **auto** mode immediately runs the `aws` CLI command that creates the OIDC provider.

6. Check the status of your cluster:

   ```bash
   $ rosa describe cluster --cluster <cluster_name|cluster_id>  
   ``

**Example output**

| Name:      | <cluster_name> |
| ID:        | <cluster_id>   |
| External ID:| <external_id>  |
| OpenShift Version: | <version> |
| Channel Group: | stable |
| DNS:       | <cluster_name>.xxxx.p1.openshiftapps.com |
| AWS Account: | <aws_account_id> |
| API URL:   | https://api.<cluster_name>.xxxx.p1.openshiftapps.com:6443 |
| Console URL: | https://console-openshift-console.apps.<cluster_name>.xxxx.p1.openshiftapps.com |
| Region:    | <aws_region>   |
| Multi-AZ:  | false          |
| Nodes:     |                |
| - Master:  | 3              |
| - Infra:   | 2              |
| - Compute: | 2              |
The following State field changes are listed in the output as the cluster installation progresses:

- **waiting** (Waiting for OIDC configuration)
- **pending** (Preparing account)
- **installing** (DNS setup in progress)
- **installing**
- **ready**

**NOTE**

If installation fails or the State field does not change to **ready** after about 40 minutes, check the installation troubleshooting documentation for more details.

7. Track the progress of the cluster creation by watching the OpenShift installer logs:

```bash
$ rosa logs install --cluster <cluster_name|cluster_id> --watch
```

Specify the `--watch` flag to watch for new log messages as the installation progresses. This argument is optional.
5.3. NEXT STEPS

- Accessing a ROSA cluster

5.4. ADDITIONAL RESOURCES

- For more information about the AWS Identity Access Management (IAM) resources required to deploy Red Hat OpenShift Service on AWS with STS, see About IAM resources for clusters that use STS.

- For an overview of the options that are presented when you create a cluster using interactive mode, see Interactive cluster creation mode reference.

- For information about the prerequisites to installing ROSA with STS, see AWS prerequisites for ROSA with STS.

- For more information about using OpenID Connect (OIDC) identity providers in AWS IAM, see Creating OpenID Connect (OIDC) identity providers in the AWS documentation.

- For more information about etcd encryption, see the etcd encryption service definition.

- For more information about troubleshooting ROSA cluster deployments, see Troubleshooting cluster deployments.
CHAPTER 6. ABOUT IAM RESOURCES FOR ROSA CLUSTERS THAT USE STS

To deploy a Red Hat OpenShift Service on AWS (ROSA) cluster that uses the AWS Security Token Service (STS), you must create the following AWS Identity Access Management (IAM) resources:

- Specific account-wide IAM roles and policies that provide the STS permissions required for ROSA support, installation, control plane, and compute functionality. This includes account-wide Operator policies.
- Cluster-specific Operator IAM roles that permit the ROSA cluster Operators to carry out core OpenShift functionality.
- An OpenID Connect (OIDC) provider that the cluster Operators use to authenticate.

This document provides reference information about the IAM resources that you must deploy when you create a ROSA cluster that uses STS. It also includes the `aws` CLI commands that are generated when you use `manual` mode with the `rosa create` command.

Additional resources

- For steps to quickly create a ROSA cluster with STS, including the AWS IAM resources, see Creating a ROSA cluster with STS quickly.
- For steps to create a ROSA cluster with STS using customizations, including the AWS IAM resources, see Creating a ROSA cluster with STS using customizations.

6.1. ACCOUNT-WIDE IAM ROLE AND POLICY REFERENCE

This section provides details about the account-wide IAM roles and policies that are required for ROSA deployments that use STS, including the Operator policies. It also includes the JSON files that define the policies.

The account-wide roles and policies are specific to an OpenShift minor release version, for example OpenShift 4.8, and are backward compatible. You can minimize the required STS resources by reusing the account-wide roles and policies for multiple clusters of the same minor version, regardless of their patch version.

**NOTE**

The account number present in the `sts_installer_trust_policy.json` and `sts_support_trust_policy.json` samples represents the Red Hat account that is allowed to assume the required roles.

Table 6.1. ROSA installer role, policy, and policy files

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedOpenShift-Installer-Role</td>
<td>An IAM role used by the ROSA installer.</td>
</tr>
<tr>
<td>ManagedOpenShift-Installer-Role-Policy</td>
<td>An inline IAM policy that provides the ROSA installer with the permissions required to complete cluster installation tasks.</td>
</tr>
<tr>
<td>Resource</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
</tbody>
</table>

**Example 6.1. sts_installer_trust_policy.json**

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

**Example 6.2. sts_installer_permission_policy.json**

```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",
"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:StartInstances",
"ec2:StopInstances",
"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"
Table 6.2. ROSA control plane role, policy, and policy files

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;route53:ListResourceRecordSets&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;route53:ListTagsForResource&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;route53:UpdateHostedZoneComment&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:CreateBucket&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:DeleteBucket&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:DeleteObject&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetAccelerateConfiguration&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetBucketAcl&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetBucketCORS&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetBucketLocation&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetBucketLogging&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetBucketObjectLockConfiguration&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetBucketRequestPayment&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetBucketTagging&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetBucketVersioning&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetBucketWebsite&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetEncryptionConfiguration&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetLifecycleConfiguration&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetObject&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetObjectAcl&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetObjectTagging&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetObjectVersion&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:GetReplicationConfiguration&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:ListBucket&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:ListBucketVersions&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:PutBucketAcl&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:PutBucketTagging&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:PutEncryptionConfiguration&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:PutObject&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:PutObjectAcl&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;s3:PutObjectTagging&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;sts:AssumeRole&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;sts:AssumeRoleWithWebIdentity&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;sts:GetCallerIdentity&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;tag:GetResources&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;tag:UntagResources&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;ec2:CreateVpcEndpointServiceConfiguration&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;ec2:DeleteVpcEndpointServiceConfigurations&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;ec2:DescribeVpcEndpointServiceConfigurations&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;ec2:DescribeVpcEndpointServicePermissions&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;ec2:DescribeVpcEndpointServices&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;ec2:ModifyVpcEndpointServicePermissions&quot;</td>
<td></td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedOpenShift-ControlPlane-Role</td>
<td>An IAM role used by the ROSA control plane.</td>
</tr>
<tr>
<td>ManagedOpenShift-ControlPlane-Role-Policy</td>
<td>An inline IAM policy that provides the ROSA control plane with the permissions required to manage its components.</td>
</tr>
</tbody>
</table>

Example 6.3. sts_instance_controlplane_trust_policy.json

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "ec2.amazonaws.com"
        ]
      },
      "Action": [
        "sts:AssumeRole"
      ]
    }
  ]
}
```

Example 6.4. sts_instance_controlplane_permission_policy.json

```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`
      ]
    }```
### Table 6.3. ROSA compute node role, policy, and policy files

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ManagedOpenShift-Worker-Role</strong></td>
<td>An IAM role used by the ROSA compute instances.</td>
</tr>
<tr>
<td><strong>ManagedOpenShift-Worker-Role-Policy</strong></td>
<td>An inline IAM policy that provides the ROSA compute instances with the permissions required to manage their components.</td>
</tr>
</tbody>
</table>

### Example 6.5. `sts_instance_worker_trust_policy.json`

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ec2.amazonaws.com"
                ],
            },
            "Action": [
                "sts:AssumeRole"
            ],
        }
    ]
}
```
Example 6.6. sts_instance_worker_permission_policy.json

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances"
            ],
            "Resource": "*"
        }
    ]
}
```

Example 6.7. sts_support_trust_policy.json

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                ],
            },
            "Action": [
                "sts:AssumeRole"
            ]
        }
    ]
}
```

Table 6.4. ROSA support role, policy, and policy files

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedOpenShift-Support-Role</td>
<td>An IAM role used by the Red Hat Site Reliability Engineering (SRE) support team.</td>
</tr>
<tr>
<td>ManagedOpenShift-Support-Role-Policy</td>
<td>An inline IAM policy that provides the Red Hat SRE support team with the permissions required to support ROSA clusters.</td>
</tr>
</tbody>
</table>
Example 6.8. sts_support_permission_policy.json

```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",
            "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",
"ec2:DescribeTransitGatewayAttachments",
"ec2:DescribeTransitGatewayConnectPeers",
"ec2:DescribeTransitGatewayConnects",
"ec2:DescribeTransitGatewayMulticastDomains",
"ec2:DescribeTransitGatewayPeeringAttachments",
"ec2:DescribeTransitGatewayRouteTables",
"ec2:DescribeTransitGatewayVpcAttachments",
"ec2:DescribeTransitGateways",
"ec2:DescribeVolumeAttribute",
"ec2:DescribeVolumeStatus",
"ec2:DescribeVolumes",
"ec2:DescribeVolumesModifications",
"ec2:DescribeVpcAttribute",
"ec2:DescribeVpcClassicLink",
"ec2:DescribeVpcClassicLinkDnsSupport",
"ec2:DescribeVpcEndpointConnectionNotifications",
"ec2:DescribeVpcEndpointConnections",
"ec2:DescribeVpcEndpointServiceConfigurations",
"ec2:DescribeVpcEndpointServicePermissions",
"ec2:DescribeVpcEndpointServices",
"ec2:DescribeVpcEndpoints",
"ec2:DescribeVpcPeeringConnections",
"ec2:DescribeVpcs",
"ec2:DescribeVpnConnections",
"ec2:DescribeVpnGateways",
"ec2:GetAssociatedIpv6PoolCidrs",
"ec2:GetTransitGatewayAttachmentPropagations",
"ec2:GetTransitGatewayMulticastDomainAssociations",
"ec2:GetTransitGatewayPrefixListReferences",
"ec2:GetTransitGatewayRouteTableAssociations",
"ec2:GetTransitGatewayRouteTablePropagations",
"ec2:RebootInstances",
"ec2:SearchLocalGatewayRoutes",
"ec2:SearchTransitGatewayMulticastGroups",
"ec2:SearchTransitGatewayRoutes",
"ec2:RunInstances",
"ec2:StartInstances",
"ec2:StopInstances",
"ec2:TerminateInstances"
Table 6.5. ROSA Ingress Operator IAM policy and policy file

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedOpenShift-openshift-ingress-operator-cloud-credentials</td>
<td>A managed IAM policy that provides the ROSA Ingress Operator with the permissions required to manage external access to a cluster.</td>
</tr>
</tbody>
</table>

Example 6.9. openshift_ingress_operator_cloud_credentials_policy.json

```json
{
    "Version": "2012-10-17",
    "elasticloadbalancing:ConfigureHealthCheck",
    "elasticloadbalancing:DescribeAccountLimits",
    "elasticloadbalancing:DescribeInstanceHealth",
    "elasticloadbalancing:DescribeListenerCertificates",
    "elasticloadbalancing:DescribeListeners",
    "elasticloadbalancing:DescribeLoadBalancerAttributes",
    "elasticloadbalancing:DescribeLoadBalancerAttributes",
    "elasticloadbalancing:DescribeLoadBalancerAttributes",
    "elasticloadbalancing:DescribeLoadBalancerPolicyTypes",
    "elasticloadbalancing:DescribeLoadBalancers",
    "elasticloadbalancing:DescribeLoadBalancers",
    "elasticloadbalancing:DescribeLoadBalancers",
    "elasticloadbalancing:DescribeSSLPolicies",
    "elasticloadbalancing:DescribeTags",
    "elasticloadbalancing:DescribeTags",
    "elasticloadbalancing:DescribeTargetGroupAttributes",
    "elasticloadbalancing:DescribeTargetGroups",
    "elasticloadbalancing:DescribeTargetHealth",
    "route53:GetHostedZone",
    "route53:GetHostedZoneCount",
    "route53:ListHostedZones",
    "route53:ListHostedZonesByName",
    "route53:ListResourceRecordSets",
    "s3:GetBucketTagging",
    "s3:GetObjectAcl",
    "s3:GetObjectTagging",
    "s3:ListAllMyBuckets"
}
```
Table 6.6. ROSA back-end storage IAM policy and policy file

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedOpenShift-openshift-cluster-csi-drivers-ebs-cloud-credentials</td>
<td>A managed IAM policy required by ROSA to manage back-end storage through the Container Storage Interface (CSI).</td>
</tr>
</tbody>
</table>

Example 6.10. openshift_cluster_csi_drivers_ebs_cloud_credentials_policy.json

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "elasticloadbalancing:DescribeLoadBalancers",
        "route53:ListHostedZones",
        "route53:ChangeResourceRecordSets",
        "tag:GetResources"
      ],
      "Resource": "*"
    }
  ]
}
```
Table 6.7. ROSA Machine Config Operator policy and policy file

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedOpenShift-openshift-machine-api-aws-cloud-credentials</td>
<td>A managed IAM policy that provides the ROSA Machine Config Operator with the permissions required to perform core cluster functionality.</td>
</tr>
</tbody>
</table>

Example 6.11. openshift_machine_api_aws_cloud_credentials_policy.json

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
    ],
    "Resource": "*",
  },
  {
    "Effect": "Allow",
    "Action": ["kms:Decrypt", "kms:Encrypt", "kms:GenerateDataKey", "kms:GenerateDataKeyWithoutPlainText", "kms:DescribeKey"
    ],
    "Resource": "*",
  },
  {
    "Effect": "Allow",
    "Action": ["kms:RevokeGrant", "kms:CreateGrant",
    ]
  }
}
```
Table 6.8. ROSA Cloud Credential Operator policy and policy file

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedOpenShift-openshift-cloud-credential-operator-cloud-credentials</td>
<td>A managed IAM policy that provides the ROSA Cloud Credential Operator with the permissions required to manage cloud provider credentials.</td>
</tr>
</tbody>
</table>

Example 6.12. openshift_cloud Credential Operator_cloudcredential_operator_iam_ro_creds_policy.json

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetUser",
        "iam:GetUserPolicy",
        "iam:ListAccessKeys"
      ],
      "Resource": "*"
    }
  ]
}
```

Table 6.9. ROSA Image Registry Operator policy and policy file

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedOpenShift-openshift-image-registry-installer-cloud-credentials</td>
<td>A managed IAM policy that provides the ROSA Image Registry Operator with the permissions required to manage the internal registry storage in AWS S3 for a cluster.</td>
</tr>
</tbody>
</table>

Example 6.13. openshift_image_registry_installer_cloud_credentials_policy.json

```json
{
  "kms:ListGrants"
}
```

"Resource": "*",
"Condition": {
  "Bool": {
    "kms:GrantIsForAWSResource": true
  }
}
```
Additional resources

- For a definition of OpenShift major, minor, and patch versions, see the Red Hat OpenShift Service on AWS update life cycle.

### 6.1.1. Account-wide IAM role and policy AWS CLI reference

This section lists the `aws` CLI commands that are shown in the terminal when you run the following `rosa` command using `manual` mode:

```
$ rosa create account-roles --mode manual
```

**NOTE**

When using `manual` mode, the `aws` commands are printed to the terminal for your review. After reviewing the `aws` commands, you must run them manually. Alternatively, you can specify `--mode auto` with the `rosa create` command to run the `aws` commands immediately.

**Command output**

```
aws iam create-role \
  --role-name ManagedOpenShift-Installer-Role \
  --assume-role-policy-document file://sts_installer_trust_policy.json \
  --tags Key=rosa_openshift_version,Value=4.8 Key=rosa_role_prefix,Value=ManagedOpenShift \
  Key=rosa_role_type,Value=installer
aws iam put-role-policy \
```

"Version": "2012-10-17",
"Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:DeleteBucket",
        "s3:PutBucketTagging",
        "s3:PutBucketPublicAccessBlock",
        "s3:PutEncryptionConfiguration",
        "s3:PutLifecycleConfiguration",
        "s3:GetBucketLocation",
        "s3:ListBucket",
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject",
        "s3:AbortMultipartUpload",
        "s3:ListMultipartUploadParts"
      ],
      "Resource": "*"
    }
  ]
aws iam create-role
--role-name ManagedOpenShift-Installer-Role \
--policy-name ManagedOpenShift-Installer-Role-Policy \
--policy-document file://sts_installer_permission_policy.json

aws iam put-role-policy
--role-name ManagedOpenShift-Installer-Role \
--policy-name ManagedOpenShift-Installer-Role-Policy \
--policy-document file://sts_installer_permission_policy.json

aws iam create-role
--role-name ManagedOpenShift-ControlPlane-Role \
--assume-role-policy-document file://sts_instance_controlplane_trust_policy.json \
--tags Key=rosa_openshift_version,Value=4.8 Key=rosa_role_prefix,Value=ManagedOpenShift 
Key=rosa_role_type,Value=instance_controlplane

aws iam put-role-policy
--role-name ManagedOpenShift-ControlPlane-Role \
--policy-name ManagedOpenShift-ControlPlane-Role-Policy \
--policy-document file://sts_instance_controlplane_permission_policy.json

aws iam create-role
--role-name ManagedOpenShift-Worker-Role \
--assume-role-policy-document file://sts_instance_worker_trust_policy.json \
--tags Key=rosa_openshift_version,Value=4.8 Key=rosa_role_prefix,Value=ManagedOpenShift 
Key=rosa_role_type,Value=instance_worker

aws iam put-role-policy
--role-name ManagedOpenShift-Worker-Role \
--policy-name ManagedOpenShift-Worker-Role-Policy \
--policy-document file://sts_instance_worker_permission_policy.json

aws iam create-role
--role-name ManagedOpenShift-Support-Role \
--assume-role-policy-document file://sts_support_trust_policy.json \
--tags Key=rosa_openshift_version,Value=4.8 Key=rosa_role_prefix,Value=ManagedOpenShift 
Key=rosa_role_type,Value=support

aws iam put-role-policy
--role-name ManagedOpenShift-Support-Role \
--policy-name ManagedOpenShift-Support-Role-Policy \
--policy-document file://sts_support_permission_policy.json

aws iam create-policy
--policy-name ManagedOpenShift-openshift-ingress-operator-cloud-credentials \
--policy-document file://openshift_ingress_operator_cloud_credentials_policy.json \
--tags Key=rosa_openshift_version,Value=4.8 Key=rosa_role_prefix,Value=ManagedOpenShift 
Key=operator_namespace,Value=openshift-ingress-operator Key=operator_name,Value=cloud-credentials

aws iam create-policy
--policy-name ManagedOpenShift-openshift-cluster-csi-drivers-ebs-cloud-credentials \
--policy-document file://openshift_cluster_csi_drivers_ebs_cloud_credentials_policy.json \
--tags Key=rosa_openshift_version,Value=4.8 Key=rosa_role_prefix,Value=ManagedOpenShift 
Key=operator_namespace,Value=openshift-cluster-csi-drivers Key=operator_name,Value=ebs-cloud-credentials

aws iam create-policy
--policy-name ManagedOpenShift-openshift-machine-api-aws-cloud-credentials \
--policy-document file://openshift_machine_api_aws_cloud_credentials_policy.json \
--tags Key=rosa_openshift_version,Value=4.8 Key=rosa_role_prefix,Value=ManagedOpenShift 
Key=operator_namespace,Value=openshift-machine-api Key=operator_name,Value=aws-cloud-
### 6.2. CLUSTER-SPECIFIC OPERATOR IAM ROLE REFERENCE

This section provides details about the Operator IAM roles that are required for ROSA deployments that use STS.

When you create the Operator roles by using the `rosa` CLI, the account-wide Operator policies for the matching cluster version are attached to the roles. The Operator policies are tagged with the Operator and version they are compatible with. The correct policy for an Operator role is determined by using the tags.

**NOTE**

If more than one matching policy is available in your account for an Operator role, an interactive list of options is provided when you create the Operator.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedOpenShift-openshift-cluster-csi-drivers-ebs-cloud-credentials</td>
<td>An IAM role required by ROSA to manage back-end storage through the Container Storage Interface (CSI).</td>
</tr>
<tr>
<td>ManagedOpenShift-openshift-machine-api-aws-cloud-credentials</td>
<td>An IAM role required by the ROSA Machine Config Operator to perform core cluster functionality.</td>
</tr>
</tbody>
</table>

The command examples provided in the table include the `ManagedOpenShift` prefix. The prefix is implied if you do not specify a custom prefix by using the `--prefix` option.
<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedOpenShift-openshift-cloud-credential-operator-cloud-credentials</td>
<td>An IAM role required by the ROSA Cloud Credential Operator to cloud provider credentials.</td>
</tr>
<tr>
<td>ManagedOpenShift-openshift-image-registry-installer-cloud-credentials</td>
<td>An IAM role required by the ROSA Image Registry Operator to manage the internal registry storage in AWS S3 for a cluster.</td>
</tr>
<tr>
<td>ManagedOpenShift-openshift-ingress-operator-cloud-credentials</td>
<td>An IAM role required by the ROSA Ingress Operator to manage external access to a cluster.</td>
</tr>
</tbody>
</table>

### 6.2.1. Operator IAM role AWS CLI reference

This section lists the `aws` CLI commands that are shown in the terminal when you run the following `rosa` command using `manual` mode:

```bash
$ rosa create operator-roles --mode manual --cluster <cluster_name>
```

**NOTE**

When using `manual` mode, the `aws` commands are printed to the terminal for your review. After reviewing the `aws` commands, you must run them manually. Alternatively, you can specify `--mode auto` with the `rosa create` command to run the `aws` commands immediately.

**Command output**

```bash
aws iam create-role \
--role-name <cluster_name>-xxxx-openshift-cluster-csi-drivers-ebs-cloud-credent \
--assume-role-policy-document file://operator_cluster_csi_drivers_ebs_cloud_credentials_policy.json \
--tags Key=rosa_cluster_id,Value=<id> Key=rosa_openshift_version,Value=4.8 Key=rosa_role_prefix,Value= Key=operator_namespace,Value=openshift-cluster-csi-drivers Key=operator_name,Value=ebs-cloud-credentials

aws iam attach-role-policy \
--role-name <cluster_name>-xxxx-openshift-cluster-csi-drivers-ebs-cloud-credent \
--policy-arn arn:aws:iam::<aws_account_id>:policy/ManagedOpenShift-openshift-cluster-csi-drivers-ebs-cloud-credent

aws iam create-role \
--role-name <cluster_name>-xxxx-openshift-machine-api-aws-cloud-credentials \
--assume-role-policy-document file://operator_machine_api_aws_cloud_credentials_policy.json \
--tags Key=rosa_cluster_id,Value=<id> Key=rosa_openshift_version,Value=4.8 Key=rosa_role_prefix,Value= Key=operator_namespace,Value=openshift-machine-api Key=operator_name,Value=aws-cloud-credentials

aws iam attach-role-policy \
--role-name <cluster_name>-xxxx-openshift-machine-api-aws-cloud-credentials \
```
NOTE

The command examples provided in the table include Operator roles that use the ManagedOpenShift prefix. If you defined a custom prefix when you created the account-wide roles and policies, including the Operator policies, you must reference it by using the --prefix <prefix_name> option when you create the Operator roles.

6.3. OIDC PROVIDER REQUIREMENTS FOR OPERATOR AUTHENTICATION

For ROSA installations that use STS, you must create a cluster-specific OIDC provider that is used by the cluster Operators to authenticate.
6.3.1. OIDC provider AWS CLI reference

This section lists the `aws` CLI command that is shown in the terminal when you run the following `rosa` command using `manual` mode:

```
$ rosa create oidc-provider --mode manual --cluster <cluster_name>
```

**NOTE**

When using manual mode, the `aws` command is printed to the terminal for your review. After reviewing the `aws` command, you must run it manually. Alternatively, you can specify `--mode auto` with the `rosa create` command to run the `aws` command immediately.

**Command output**

```
aws iam create-open-id-connect-provider \
--url https://rh-oidc.s3.<aws_region>.amazonaws.com/<cluster_id> \
--client-id-list openshift sts.amazonaws.com \
--thumbprint-list <thumbprint> 1
```

1 The thumbprint is generated automatically when you run the `rosa create oidc-provider` command. For more information about using thumbprints with AWS Identity and Access Management (IAM) OpenID Connect (OIDC) identity providers, see the AWS documentation.
CHAPTER 7. INTERACTIVE CLUSTER CREATION MODE REFERENCES

This section provides an overview of the options that are presented when you use the interactive mode to create a cluster through the rosa CLI.

7.1. UNDERSTANDING THE INTERACTIVE CLUSTER CREATION MODE OPTIONS

You can create a Red Hat OpenShift Service on AWS cluster with the AWS Security Token Service (STS) by using the interactive mode. You can enable the mode by specifying the --interactive option when you run rosa create cluster. The following table describes the interactive mode options.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster name</td>
<td>Enter a name for your cluster, for example my-rosa-cluster.</td>
</tr>
<tr>
<td>Deploy cluster using AWS STS</td>
<td>Create an OpenShift cluster that uses the AWS Security Token Service (STS) to allocate temporary, limited-privilege credentials for component-specific AWS Identity and Access Management (IAM) roles. The service enables cluster components to make AWS API calls using secure cloud resource management practices.</td>
</tr>
<tr>
<td>OpenShift version</td>
<td>Select the version of OpenShift to install, for example 4.3.12. The default is the latest version.</td>
</tr>
<tr>
<td>External ID (optional)</td>
<td>Specify an unique identifier that is passed by OpenShift Cluster Manager and the OpenShift installer when an account role is assumed. This option is only required for custom account roles that expect an external ID.</td>
</tr>
<tr>
<td>Operator roles prefix</td>
<td>Enter a prefix to assign to the cluster-specific Operator IAM roles. The default is the name of the cluster and a 4-digit random string, for example my-rosa-cluster-a0b1.</td>
</tr>
<tr>
<td>Multiple availability zones</td>
<td>Deploy the cluster to multiple availability zones in the AWS region. The default is No, which results in a cluster being deployed to a single availability zone. If you deploy a cluster into multiple availability zones, the AWS region must have at least 3 availability zones. Multiple availability zones are recommended for production workloads.</td>
</tr>
<tr>
<td>AWS region</td>
<td>Specify the AWS region to deploy the cluster in. This overrides the AWS_REGION environment variable.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PrivateLink cluster</td>
<td>Create a cluster using AWS PrivateLink. This option provides private connectivity between Virtual Private Clouds (VPCs), AWS services, and your on-premise networks, without exposing your traffic to the public internet. To provide support, Red Hat Site Reliability Engineering (SRE) can connect to the cluster by using AWS PrivateLink Virtual Private Cloud (VPC) endpoints. This option cannot be changed after a cluster is created. The default is No.</td>
</tr>
<tr>
<td>Install into an existing VPC</td>
<td>Install a cluster into an existing AWS VPC. To use this option, your VPC must have 2 subnets for each availability zone that you are installing the cluster into. The default is No.</td>
</tr>
<tr>
<td>Enable customer managed key</td>
<td>Enable this option to use the AWS Key Management Service (KMS) to help securely manage keys for encrypted data. The keys are used for control plane data volumes that are encrypted by default. Persistent volumes (PVs) for customer applications also use AWS KMS for key management. When enabled, the account KMS key for the region is used by default. The default is No.</td>
</tr>
<tr>
<td>Compute nodes instance type</td>
<td>Select a compute node instance type. The default is m5.xlarge.</td>
</tr>
<tr>
<td>Enable autoscaling</td>
<td>Enable compute node autoscaling. The autoscaler adjusts the size of the cluster to meet your deployment demands. The default is No.</td>
</tr>
<tr>
<td>Compute nodes</td>
<td>Specify the number of compute nodes to provision into each availability zone. Clusters deployed in a single availability zone require at least 2 nodes. Clusters deployed in multiple zones must have at least 3 nodes. The default is 2.</td>
</tr>
<tr>
<td>Machine CIDR</td>
<td>Specify the IP address range for machines (cluster nodes), which must encompass all CIDR address ranges for your VPC subnets. Subnets must be contiguous. A minimum IP address range of 128 addresses, using the subnet prefix /25, is supported for single availability zone deployments. A minimum address range of 256 addresses, using the subnet prefix /24, is supported for deployments that use multiple availability zones. The default is 10.0.0.0/16. This range must not conflict with any connected networks.</td>
</tr>
<tr>
<td>Service CIDR</td>
<td>Specify the IP address range for services. The range must be large enough to accommodate your workload. The address block must not overlap with any external service accessed from within the cluster. The default is 172.30.0.0/16. It is recommended that they are the same between clusters.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Pod CIDR</td>
<td>Specify the IP address range for pods. The range must be large enough to accommodate your workload. The address block must not overlap with any external service accessed from within the cluster. The default is 10.128.0.0/14. It is recommended that they are the same between clusters.</td>
</tr>
<tr>
<td>Host prefix</td>
<td>Specify the subnet prefix length assigned to pods scheduled to individual machines. The host prefix determines the pod IP address pool for each machine. For example, if the host prefix is set to /23, each machine is assigned a /23 subnet from the pod CIDR address range. The default is /23, allowing 512 cluster nodes and 512 pods per node, both of which are beyond our supported maximums. For information on the supported maximums, see the Additional resources section below.</td>
</tr>
<tr>
<td>Encrypt etcd data (optional)</td>
<td>In Red Hat OpenShift Service on AWS, the control plane storage is encrypted at rest by default and this includes encryption of the etcd volumes. You can additionally enable the Encrypt etcd data option to encrypt the key values for some resources in etcd, but not the keys. IMPORTANT By enabling etcd encryption for the key values in etcd, you will incur a performance overhead of approximately 20%. The overhead is a result of introducing this second layer of encryption, in addition to the default control plane storage encryption that encrypts the etcd volumes. Red Hat recommends that you enable etcd encryption only if you specifically require it for your use case.</td>
</tr>
<tr>
<td>Disable workload monitoring</td>
<td>Disable monitoring for user-defined projects. Monitoring for user-defined projects is enabled by default.</td>
</tr>
</tbody>
</table>

### 7.2. ADDITIONAL RESOURCES

- For a list of the supported maximums, see [ROSA tested cluster maximums](#).
- For detailed steps to quickly create a ROSA cluster with STS, including the AWS IAM resources, see [Creating a ROSA cluster with STS quickly](#).
- For detailed steps to create a ROSA cluster with STS using customizations, including the AWS IAM resources, see [Creating a ROSA cluster with STS using customizations](#).
- For more information about etcd encryption, see the [etcd encryption service definition](#).
- For an example VPC architecture, see [this sample VPC architecture](#).
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.

This document describes how to access a cluster and set up an IDP using the `rosa` CLI. Alternatively, you can create an IDP account using OpenShift Cluster Manager console.

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-2mkJl-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-2mkJl-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

   -
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&oauth_application%5Bname%5D=rh-rosa-test-cluster-
       stage&oath_application%5Burl%5D=https%3A%2F%2Fconsole-openshift-
       console.apps.rh-rosa-test-cluster.z7v0.s1.devshift.org
     - Click on 'Register application'
     ...
   ```

   c. 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 that is defined by the rosa CLI tool.

d. 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**

```
... 
? 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.

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

```
$ 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:

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

**Example output**

```
Name:        rh-rosa-test-cluster1 
ID:          1de87g7c30g75qechgh7l5b2bha6r04e
```
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.

   ```bash
   $ 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.

   ```bash
   $ 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:

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

2. Verify your user is listed as a cluster administrator:

   $ 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.

   $ 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>
<tr>
<td>NAME</td>
<td>TYPE</td>
<td>CLUSTER-IP</td>
<td>EXTERNAL-IP</td>
<td>PORT(S)</td>
</tr>
<tr>
<td>service/api</td>
<td>ClusterIP</td>
<td>172.30.23.241</td>
<td>&lt;none&gt;</td>
<td>443/TCP</td>
</tr>
<tr>
<td>NAME</td>
<td>DESIRED</td>
<td>CURRENT</td>
<td>READY</td>
<td>UP-TO-DATE</td>
</tr>
<tr>
<td>daemonset.apps/apiserver</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</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:

   `$ 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:

   `$ 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.

8.5. ADDITIONAL RESOURCES

- Configuring identity providers using Red Hat OpenShift Cluster Manager console
- Understanding the ROSA with STS deployment workflow
CHAPTER 9. 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.

The following topics describe how to configure an identity provider using OpenShift Cluster Manager console. Alternatively, you can use the `rosa` CLI to configure an identity provider and access the cluster.

9.1. 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.

9.1.1. 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 GitHub identity provider to validate usernames and passwords against GitHub or GitHub Enterprise’s OAuth authentication server.</td>
</tr>
<tr>
<td>GitLab</td>
<td>Configure a GitLab identity provider to use GitLab.com or any other GitLab instance as an identity provider.</td>
</tr>
<tr>
<td>Google</td>
<td>Configure a Google identity provider using Google’s OpenID Connect integration.</td>
</tr>
<tr>
<td>LDAP</td>
<td>Configure an LDAP identity provider to validate usernames and passwords against an LDAPv3 server, using simple bind authentication.</td>
</tr>
<tr>
<td>OpenID Connect</td>
<td>Configure an OpenID Connect (OIDC) identity provider to integrate with an OIDC identity provider using an Authorization Code Flow.</td>
</tr>
<tr>
<td>HTPasswd</td>
<td>Configure an HTPasswd identity provider for a single, static administration user. You can log in to the cluster as the user to troubleshoot issues.</td>
</tr>
</tbody>
</table>

**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. For the steps to configure the single user, see Configuring an HTPasswd identity provider.

9.1.2. 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>
| mappingMethod | 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`.

### 9.2. 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. From OpenShift Cluster Manager, 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.
9.3. 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. From OpenShift Cluster Manager, 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.

## 9.4. 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. From **OpenShift Cluster Manager**, 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.

9.5. 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:


<table>
<thead>
<tr>
<th>URL component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldap</td>
<td>For regular LDAP, use the string <code>ldap</code>. For secure LDAP (LDAPS), use <code>ldaps</code> instead.</td>
</tr>
<tr>
<td>host:port</td>
<td>The name and port of the LDAP server. Defaults to <code>localhost:389</code> for ldap and <code>localhost:636</code> 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 <code>uid</code>. 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 <code>one</code> or <code>sub</code>. If the scope is not provided, the default is to use a scope of <code>sub</code>.</td>
</tr>
<tr>
<td>filter</td>
<td>A valid LDAP search filter. If not provided, defaults to <code>(objectClass=*).</code></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. From OpenShift Cluster Manager, 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
9.6. 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><code>preferred_username</code></td>
<td>The preferred user name when provisioning a user. A shorthand name that the user wants to be referred to as, such as <code>janedoe</code>. Typically a value that corresponding to the user's login or username in the authentication system, such as username or email.</td>
</tr>
<tr>
<td><code>email</code></td>
<td>Email address.</td>
</tr>
<tr>
<td><code>name</code></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. From OpenShift Cluster Manager, 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.
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.

9.7. CONFIGURING AN HTPASSWD IDENTITY PROVIDER

Configure an HTPasswd identity provider to create a single, static user with cluster administration privileges. You can log in to your cluster as the user to troubleshoot issues.
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.

Procedure

1. From OpenShift Cluster Manager, navigate to the Clusters page and select your cluster.
2. Select Access control → Identity providers.
3. Click Add identity provider.
4. Select HTPasswd from the Identity Provider drop-down menu.
5. Add a unique name in the Name field for the identity provider.
6. Use the suggested username and password for the static user, or create your own.

   NOTE

The credentials defined in this step are not visible after you select Add in the following step. If you lose the credentials, you must recreate the identity provider and define the credentials again.

7. Select Add to create the HTPasswd identity provider and the single, static user.

8. Grant the static user permission to manage the cluster:
   a. Under Access control → Cluster Roles and Access, select Add user.
   b. Enter the User ID of the static user that you created in the preceding step.
   c. Select a Group. Users in the dedicated-admins group have standard administrative privileges for Red Hat OpenShift Service on AWS. Users in the cluster-admins group have full administrative access to the cluster.
   d. Select Add user to grant the administration privileges to the user.

Verification

- The configured HTPasswd identity provider is visible on the Access control → Identity providers page.

   NOTE

After creating the identity provider, synchronization usually completes within two minutes. You can log in to the cluster as the user after the HTPasswd identity provider becomes available.

- The single, administrative user is visible on the Access control → Cluster Roles and Access page. The administration group membership of the user is also displayed.
9.8. ADDITIONAL RESOURCES

- Accessing a cluster
- Understanding the ROSA with STS deployment workflow
CHAPTER 10. REVOKING ACCESS TO A ROSA CLUSTER

An identity provider (IDP) controls access to a Red Hat OpenShift Service on AWS (ROSA) cluster. To revoke access of a user to a cluster, you must configure that within the IDP that was set up for authentication.

10.1. REVOKING ADMINISTRATOR ACCESS USING THE ROSA CLI

You can revoke the administrator access of users so that they can access the cluster without administrator privileges. To remove the administrator access for a user, you must revoke the dedicated-admin or cluster-admin privileges. You can revoke the administrator privileges using the rosa command-line utility or using OpenShift Cluster Manager console.

10.1.1. Revoking dedicated-admin access using the rosa CLI

You can revoke access for a dedicated-admin user if you are the user who created the cluster, the organization administrator user, or the super administrator user.

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 the dedicated-admin access of a user:

   $ 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 output does not list the revoked user.

   $ oc get groups dedicated-admins

10.1.2. Revoking cluster-admin access using the rosa CLI

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. Enter the following command to revoke the cluster-admin access of a user:
$ rosa revoke user cluster-admins --user=myusername --cluster=mycluster

2. Enter the following command to verify that the user no longer has `cluster-admin` access. The output does not list the revoked user.

$ oc get groups cluster-admins

10.2. REVOKING ADMINISTRATOR ACCESS USING OPENSHIFT CLUSTER MANAGER CONSOLE

You can revoke the `dedicated-admin` or `cluster-admin` access of users through OpenShift Cluster Manager console. Users will be able to access the cluster without administrator privileges.

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 OpenShift Cluster Manager console using an OpenShift Cluster Manager account that you used to create the cluster, the organization administrator user, or the super administrator user.

Procedure

1. On the Clusters tab of OpenShift Cluster Manager, select the name of your cluster to view the cluster details.


3. For the user that you want to remove, click the Options menu to the right of the user and group combination and click Delete.
CHAPTER 11. DELETING A ROSA CLUSTER

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

11.1. PREREQUISITES

- If Red Hat OpenShift Service on AWS created a VPC, you must remove the following items from your cluster before you can successfully delete your cluster:
  - Network configurations, such as VPN configurations and VPC peering connections
  - Any additional services that were added to the VPC

If these configurations and services remain, the cluster does not delete properly.

11.2. DELETING A CLUSTER

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

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. The cluster deletion must complete before you remove the IAM resources, because the resources are used in the cluster deletion and clean-up processes.

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.

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.

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
   ```

   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:

   ```bash
   $ rosa delete oidc-provider -c <cluster_id> --mode auto
   ```

   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:

   ```bash
   $ rosa delete operator-roles -c <cluster_id> --mode auto
   ```

   Replace `<cluster_id>` with the ID of the cluster.

4. Delete the account-wide roles:

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

   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. Use the AWS IAM Console to delete the account-wide inline and Operator policies. For detailed steps, see the *Deleting the AWS resources by using the AWS IAM Console* section.

### 11.3. DELETING THE AWS RESOURCES BY USING THE AWS IAM CONSOLE

After deleting a Red Hat OpenShift Service on AWS (ROSA) cluster, you can delete the AWS Security Token Service (STS) resources by using the AWS Identity and Access Management (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 have deleted your ROSA cluster. For more information, see the *Deleting a cluster* section.

**IMPORTANT**

You must delete the cluster before you remove the IAM roles and policies. 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.
Procedure

1. Log in to the AWS IAM Console.

2. Delete the OpenID Connect (OIDC) provider that you created for Operator authentication in your cluster:
   a. Navigate to Access management → Identity providers and click on the OIDC resource that you created to authenticate the cluster Operators.
   b. In the dialog page for the resource, select **Delete** to delete the OIDC provider.

3. Delete the cluster-specific Operator IAM roles:
   
   **TIP**
   
   The IAM role and policy names include the role prefix that is specified when the STS resources are created. The default prefix is ManagedOpenShift.
   
   a. Navigate to Access management → Roles and click on one of the cluster-specific Operator roles that you created for your cluster.
   b. In the dialog page for the resource, select **Delete role** to delete the role. Select **Yes, delete** to confirm the role deletion.
   c. Repeat this step to delete each of the cluster-specific Operator roles for the cluster.

4. Delete the account-wide Operator policies that you created for ROSA deployments that use STS:
   
   a. Navigate to Access management → Policies and click on one of the Operator policies.
   b. In the dialog page for the resource, select **Delete policy** to delete the policy. Select **Delete** to confirm the policy deletion.
   c. Repeat this step to delete each of the Operator policies.

5. Delete the account-wide IAM roles and inline policies that you created for ROSA deployments that use STS:
   
   a. Navigate to Access management → Roles and click on one of the account-wide roles.
   b. In the dialog page for the resource, select **Delete role** to delete the role. Select **Yes, delete** to confirm the role deletion.
   c. Repeat this step to delete each of the account-wide roles for the cluster.
CHAPTER 12. DEPLOYING ROSA WITHOUT AWS STS

12.1. UNDERSTANDING THE ROSA DEPLOYMENT WORKFLOW

Before you create a Red Hat OpenShift Service on AWS (ROSA) cluster that uses the AWS Security Token Service (STS), 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.

12.1.1. Overview of the ROSA deployment workflow

You can follow the workflow stages outlined in this section to set up and access a Red Hat OpenShift Service on AWS (ROSA) cluster.

1. Perform the AWS prerequisites. To deploy a ROSA cluster, 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. Configure your AWS account. Before you create a ROSA cluster, you must enable ROSA in your AWS account, install and configure the AWS CLI (aws) tool, and verify the AWS CLI tool configuration.

4. Install the ROSA and OpenShift CLI tools and verify the AWS service quotas. Install and configure the ROSA CLI (aws) and the OpenShift CLI (oc). You can verify if the required AWS resource quotas are available by using the ROSA CLI.

5. Create a ROSA cluster or Create a ROSA cluster using AWS PrivateLink. Use the ROSA CLI (rosa) to create a cluster. You can optionally create a ROSA cluster with AWS PrivateLink.

6. Access a 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.

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

8. Delete a ROSA cluster. You can delete a ROSA cluster by using the ROSA CLI (rosa).

12.1.2. Additional resources

- For information about using the ROSA deployment workflow to create a cluster that uses the AWS Security Token Service (STS), see Understanding the ROSA with STS deployment workflow.

12.2. 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.

12.2.1. Required AWS service quotas
The table below describes the AWS service quotas and levels required to create and run a 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>50</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>50</td>
<td>300</td>
</tr>
<tr>
<td>Application Load Balancers per Region</td>
<td>elasticloadbalancing</td>
<td>L-53DA6B97</td>
<td>50</td>
<td>50</td>
</tr>
</tbody>
</table>
12.2.2. Next steps

- Configure your AWS account

12.2.3. Additional resources

- Understanding the ROSA deployment workflow

12.3. CONFIGURING YOUR AWS ACCOUNT

After you complete the AWS prerequisites, configure your AWS account and enable the Red Hat OpenShift Service on AWS (ROSA) service.

12.3.1. Configuring your AWS account

To configure your AWS account to use the ROSA service, complete the following steps.

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. A dedicated AWS account is recommended 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, see AWS Prerequisites for details on the minimum required SCP.

   As part of the cluster creation process, `rosa` establishes an `osdCcsAdmin` IAM user. This user uses the IAM credentials you provide when configuring the AWS CLI.

   **NOTE**

   This user has Programmatic access enabled and the AdministratorAccess policy attached to it.

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]
      It is recommended to set the default AWS region by using the environment variable.

      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
      ```

      **Example output**

      ```
      +-------------------------------+-----------------------+-----------------+
      | GetCallerIdentity             |
      +-------------------------------+-----------------------+-----------------+
      | Account | Arn | UserID |
      +-------------------------------+-----------------------+-----------------+
      | <account_name>  | arn:aws:iam<string>:user:name | <userID> |
      +-------------------------------+-----------------------+-----------------+
      ```

      After completing these steps, install ROSA.

**12.3.2. Next steps**

- Install ROSA
12.3.3. Additional resources

- AWS prerequisites
- Required AWS service quotas and requesting increases
- Understanding the ROSA deployment workflow

12.4. INSTALLING ROSA

After you configure your AWS account, install Red Hat OpenShift Service on AWS (ROSA).

12.4.1. Installing ROSA

Complete the following steps to install ROSA before creating a cluster.

Prerequisites

- Review and complete the AWS prerequisites and ROSA 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.
- Configure your AWS account and enable the ROSA service in your AWS account.

Procedure

1. Install `rosa`, the Red Hat OpenShift Service on AWS command-line interface (CLI).
   a. Download the latest release of the `rosa` CLI for your operating system.
   b. Optional: Rename the executable file you downloaded to `rosa`. This documentation uses `rosa` to refer to the executable file.
   c. Optional: Add `rosa` to your path.

   **Example**

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

   Command line tool for ROSA.

   Usage:
   ```
   rosa [command]
   ```

   Available Commands:
   ```
   completion  Generates bash completion scripts
   create      Create a resource from stdin
   ```
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
```

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
```

2. Enter the following command to verify that your AWS account has the necessary permissions.

```
$ rosa verify permissions
```

**Example output**

```
I: Validating SCP policies...
I: AWS SCP policies ok
```

3. Log in to your Red Hat account with `rosa`.

   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://console.redhat.com/openshift/token/rosa?

Copy the token and paste it here: `<my-offline-access-token>`

**Example output continued**
4. Verify that your AWS account has the necessary quota to deploy an Red Hat OpenShift Service on AWS cluster.

```bash
$ rosa verify quota --region=us-west-2
```

**Example output**

```
I: Validating AWS quota...
I: AWS quota ok
```

**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](https://console.aws.amazon.com), and request a quota increase for the service that failed.

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

5. 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.

```bash
$ rosa whoami
```

**Example output**

```
AWS Account ID:               000000000000
AWS Default Region:           us-east-2
AWS ARN:                      arn:aws:iam::000000000000:user/hello
OCM API:                      https://api.openshift.com
OCM Account ID:               1DzGldlhqEWyt8UUXQhSoWaaaaa
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
```

   b. Initialize your AWS account. This step runs a CloudFormation template that prepares your AWS account for cluster deployment and management. This step typically takes 1-2 minutes to complete.

```bash
$ rosa init
```

**Example output**
6. Install the OpenShift CLI (oc) from the rosa CLI.

   a. Enter this command to download the latest version of the oc CLI:

   ```
   $ rosa download oc
   ```

   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 oc
   ```

After installing ROSA, you are ready to create a cluster.

12.4.2. Next steps

- Create a ROSA cluster or Create an AWS PrivateLink cluster on ROSA.

12.4.3. Additional resources

- AWS prerequisites

- Required AWS service quotas and requesting increases

- Understanding the ROSA deployment workflow

12.5. CREATING A ROSA CLUSTER

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

This document describes how to set up a ROSA cluster. Alternatively, you can create a ROSA cluster with AWS PrivateLink.

12.5.1. Creating your cluster

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

Prerequisites

You have installed Red Hat OpenShift Service on AWS.
NOTE

AWS Shared VPCs are not currently supported for ROSA installs.

Procedure

1. You can create a cluster using the default settings or by specifying custom settings 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 (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 your cluster with the default cluster settings:

     $ rosa create cluster --cluster-name=<cluster_name>

   Example output

     I: Creating cluster with identifier '1de87g7c30g75qechgh7l5b2bha6r04e' and name 'rh-rosa-test-cluster1'
     I: To view list of clusters and their status, run `rosa list clusters`
     I: Cluster 'rh-rosa-test-cluster1' 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 rh-rosa-test-cluster1`.

   - To create a cluster using interactive prompts:

     $ rosa create cluster --interactive

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

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

   2. Enter the following command to check the status of your cluster. During cluster creation, the State field from the output will transition from pending to installing, and finally to ready.

     $ rosa describe cluster --cluster=<cluster_name>

   Example output
Name: rh-rosa-test-cluster1
OpenShift Version: 4.6.8
DNS: *.example.com
ID: uniqueidnumber
External ID: uniqueexternalidnumber
AWS Account: 123456789101
API URL: https://api.rh-rosa-test-cluster1.example.org:6443
Console URL: https://console-openshift-console.apps.rh-rosa-test-cluster1.example.org
Nodes: Master: 3, Infra: 2, Compute: 2
Region: us-west-2
Multi-AZ: false
State: ready
Channel Group: stable
Private: No
Created: Jan 15 2021 16:30:55 UTC
Details Page: https://console.redhat.com/examplename/details/idnumber

NOTE
If installation fails or the State field does not change to ready after 40 minutes, check the installation troubleshooting documentation for more details.

3. Track the progress of the cluster creation by watching the OpenShift installer logs:

   $ rosa logs install --cluster=<cluster_name> --watch

12.5.2. Next steps
Configure identity providers

12.5.3. Additional resources

   - Understanding the ROSA deployment workflow
   - Deleting a ROSA cluster
   - ROSA architecture

12.6. CREATING AN AWS PRIVATELINK CLUSTER ON ROSA

This document describes how to create a ROSA cluster using AWS PrivateLink. Alternatively, you can create a ROSA cluster without AWS PrivateLink.

12.6.1. Understanding AWS PrivateLink

A Red Hat OpenShift Service on AWS cluster can be created without any requirements on public subnets, internet gateways, or network address translation (NAT) gateways. In this configuration, Red Hat uses AWS PrivateLink to manage and monitor a cluster in order to avoid all public ingress network traffic.

For more information, see AWS PrivateLink on the AWS website.
12.6.2. Requirements for using AWS PrivateLink clusters

For AWS PrivateLink clusters, internet gateways, NAT gateways and public subnets are not required, but the private subnets must have internet connectivity provided to install required components. At least one single private subnet is required for Single-AZ clusters and at least 3 private subnets are required for Multi-AZ clusters. The following table shows the AWS resources that are required for a successful installation:

Table 12.1. Required AWS resources

<table>
<thead>
<tr>
<th>Component</th>
<th>AWS Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPC</td>
<td>AWS::EC2::VPC</td>
<td>You must provide a VPC for the cluster to use.</td>
</tr>
<tr>
<td></td>
<td>AWS::EC2::VPCEndpoint</td>
<td></td>
</tr>
<tr>
<td>Network access control</td>
<td>AWS::EC2::NetworkAcl</td>
<td>You must allow access to the following ports:</td>
</tr>
<tr>
<td></td>
<td>AWS::EC2::NetworkAclEntry</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port</td>
<td>Reason</td>
</tr>
<tr>
<td></td>
<td>80</td>
<td>Inbound HTTP traffic</td>
</tr>
<tr>
<td></td>
<td>443</td>
<td>Inbound HTTPS traffic</td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>Inbound SSH traffic</td>
</tr>
<tr>
<td></td>
<td>1024-65535</td>
<td>Inbound ephemeral traffic</td>
</tr>
<tr>
<td></td>
<td>0-65535</td>
<td>Outbound ephemeral traffic</td>
</tr>
<tr>
<td>Private subnets</td>
<td>AWS::EC2::Subnet</td>
<td>Your VPC must have private subnets in 1 availability zone for Single-AZ deployments or 3 availability zones for Multi-AZ deployments. You must provide appropriate routes and route tables.</td>
</tr>
<tr>
<td></td>
<td>AWS::EC2::RouteTable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AWS::EC2::SubnetRouteTableAssociation</td>
<td></td>
</tr>
</tbody>
</table>

12.6.3. Creating an AWS PrivateLink cluster

You can create an AWS PrivateLink cluster using the **rosa** CLI.

![NOTE]

AWS PrivateLink is supported on existing VPCs only.

**Prerequisites**

You have installed Red Hat OpenShift Service on AWS.
Procedure

Creating a cluster can take up to 40 minutes.

1. With AWS PrivateLink, you can create a cluster with a single availability zone (Single-AZ) or multiple availability zones (Multi-AZ). In either case, your machine’s classless inter-domain routing (CIDR) must match your virtual private cloud’s CIDR. See Requirements for using your own VPC and VPC Validation for more information.

   IMPORTANT

   If you use a firewall, you must configure it so that Red Hat OpenShift Service on AWS can access the sites that it requires to function.

   For more information, see the AWS PrivateLink firewall prerequisites section.

   • To create a Single-AZ cluster:

   ```$ rosa create cluster --private-link --cluster-name=<cluster-name> [--machine-cidr=<VPC CIDR>/16] --subnet-ids=<private-subnet-id>`

   • To create a Multi-AZ cluster:

   ```$ rosa create cluster --private-link --multi-az --cluster-name=<cluster-name> [--machine-cidr=<VPC CIDR>/16] --subnet-ids=<private-subnet-id1>,<private-subnet-id2>,<private-subnet-id3>`

2. Enter the following command to check the status of your cluster. During cluster creation, the State field from the output will transition from pending to installing, and finally to ready.

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

   NOTE

   If installation fails or the State field does not change to ready after 40 minutes, check the installation troubleshooting documentation for more details.

3. Enter the following command to follow the OpenShift installer logs to track the progress of your cluster:

   ```$ rosa logs install --cluster=<cluster_name> --watch`

12.6.4. Configuring AWS PrivateLink DNS forwarding

With AWS PrivateLink clusters, a public hosted zone and a private hosted zone are created in Route 53. With the private hosted zone, records within the zone are resolvable only from within the VPC to which it is assigned.

The Let’s Encrypt DNS-01 validation requires a public zone so that valid, publicly trusted certificates can be issued for the domain. The validation records are deleted after Let’s Encrypt validation is complete; however, the zone is still required for issuing and renewing these certificates, which are typically required every 60 days. While these zones usually appear empty, it is serving a critical role in the validation process.
For more information about private hosted zones, see AWS private hosted zones documentation. For more information about private hosted zones, see AWS public hosted zones documentation.

Prerequisites

- Your corporate network or other VPC has connectivity
- UDP port 53 and TCP port 53 are enabled across your networks to allow for DNS queries
- You have created an AWS PrivateLink cluster using Red Hat OpenShift Service on AWS

Procedure

1. To allow for records such as `api.<cluster_domain>` and `*.apps.<cluster_domain>` to resolve outside of the VPC, configure a Route 53 Resolver Inbound Endpoint.

2. When you configure the inbound endpoint, select the VPC and private subnets that were used when you created the cluster.

3. After the endpoints are operational and associated, configure your corporate network to forward DNS queries to those IP addresses for the top-level cluster domain, such as `drow-pl-01.hino.p1.openshiftapps.com`.

4. If you are forwarding DNS queries from one VPC to another VPC, configure forwarding rules.

5. If you are configuring your remote network DNS server, see your specific DNS server documentation to configure selective DNS forwarding for the installed cluster domain.

12.6.5. Next steps

Configure identity providers

12.6.6. Additional resources

- AWS PrivateLink firewall prerequisites
- Understanding the ROSA deployment workflow
- Deleting a ROSA cluster
- ROSA architecture

12.7. ACCESSING A ROSA CLUSTER

As a best practice, 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.

This document describes how to access a cluster and set up an IDP using the `rosa` CLI. Alternatively, you can set up an IDP account using OpenShift Cluster Manager console.

12.7.1. 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:

   ```bash
   $ 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:
   ```bash
   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

   ```bash
   $ 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:

   ```bash
   $ oc whoami
   ```

   Example output

   ```bash
   cluster-admin
   ```

12.7.2. 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**

   
   ```
   $ rosa create idp --cluster=example-cluster --interactive
   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&oauth_application%5Bname%5D=rh-rosa-test-cluster-stage&oauth_application%5Burl%5D=https%3A%2F%2Fconsole-openshift-console.apps.rh-rosa-test-cluster.z7v0.s1.devshift.org
   - Click on 'Register application'
   ...
   ```
   
   c. 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 that is defined by the `rosa` CLI tool.

   d. 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**

```
... 
? 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.

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

```
$ 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:

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

**Example output**

```
Name: rh-rosa-test-cluster1
ID: 1de87g7c30g75qechgh7f5b2bha6r04e
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

### 12.7.3. 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:

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

2. Verify your user is listed as a cluster administrator:

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

12.7.4. **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
   $ oc get groups dedicated-admins
   ```

**Example output**
12.7.5. Additional resources

- Configuring identity providers
- Understanding the ROSA deployment workflow

12.8. CONFIGURING IDENTITY PROVIDERS

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.

The following topics describe how to configure an identity provider using OpenShift Cluster Manager console. Alternatively, you can use the `rosa` CLI to create an identity provider and access the cluster.

12.8.1. 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.

12.8.1.1. 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 GitHub identity provider to validate usernames and passwords against GitHub or GitHub Enterprise’s OAuth authentication server.</td>
</tr>
<tr>
<td>GitLab</td>
<td>Configure a GitLab identity provider to use GitLab.com or any other GitLab instance as an identity provider.</td>
</tr>
<tr>
<td>Google</td>
<td>Configure a Google identity provider using Google’s OpenID Connect integration.</td>
</tr>
<tr>
<td>LDAP</td>
<td>Configure an LDAP identity provider to validate usernames and passwords against an LDAPv3 server, using simple bind authentication.</td>
</tr>
<tr>
<td>OpenID Connect</td>
<td>Configure an OpenID Connect (OIDC) identity provider to integrate with an OIDC identity provider using an Authorization Code Flow.</td>
</tr>
</tbody>
</table>
Configure an HTPasswd identity provider for a single, static administration user. You can log in to the cluster as the user to troubleshoot issues.

**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. For the steps to configure the single user, see Configuring an HTPasswd identity provider.

### 12.8.1.2. 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><strong>name</strong></td>
<td>The provider name is prefixed to provider user names to form an identity name.</td>
</tr>
<tr>
<td><strong>mappingMethod</strong></td>
<td>Defines how new identities are mapped to users when they log in. Enter one of the following values:</td>
</tr>
<tr>
<td>claim</td>
<td>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.</td>
</tr>
<tr>
<td>lookup</td>
<td>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.</td>
</tr>
<tr>
<td>generate</td>
<td>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.</td>
</tr>
<tr>
<td>add</td>
<td>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.</td>
</tr>
</tbody>
</table>

**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**.
12.8.2. 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. From OpenShift Cluster Manager, 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.

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.

**12.8.3. 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. From **OpenShift Cluster Manager**, 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.
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.

**12.8.4. Configuring a Google identity provider**

Configure a Google identity provider to allow users to authenticate with theirGoogle 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. From **OpenShift Cluster Manager**, 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/


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.

12.8.5. 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:


<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>URL component</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</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 <code>uid</code>. 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 <code>one</code> or <code>sub</code>. If the scope is not provided, the default is to use a scope of <code>sub</code>.</td>
</tr>
<tr>
<td>filter</td>
<td>A valid LDAP search filter. If not provided, defaults to <code>(objectClass=*)</code></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. From OpenShift Cluster Manager, 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.

12.8.6. 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></td>
<td></td>
</tr>
<tr>
<td>Claim</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<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 corresponding 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. From OpenShift Cluster Manager, 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.

**12.8.7. Configuring an HTPasswd identity provider**

Configure an HTPasswd identity provider to create a single, static user with cluster administration privileges. You can log in to your cluster as the user to troubleshoot issues.

**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.

**Procedure**

1. From **OpenShift Cluster Manager**, navigate to the **Clusters** page and select your cluster.

2. Select **Access control → Identity providers**.

3. Click **Add identity provider**.

4. Select **HTPasswd** from the **Identity Provider** drop-down menu.

5. Add a unique name in the **Name** field for the identity provider.

6. Use the suggested username and password for the static user, or create your own.

**NOTE**

The credentials defined in this step are not visible after you select **Add** in the following step. If you lose the credentials, you must recreate the identity provider and define the credentials again.
7. Select Add to create the HTPasswd identity provider and the single, static user.

8. Grant the static user permission to manage the cluster:
   a. Under Access control → Cluster Roles and Access, select Add user.
   b. Enter the User ID of the static user that you created in the preceding step.
   c. Select a Group. Users in the dedicated-admins group have standard administrative privileges for Red Hat OpenShift Service on AWS. Users in the cluster-admins group have full administrative access to the cluster.
   d. Select Add user to grant the administration privileges to the user.

Verification

- The configured HTPasswd identity provider is visible on the Access control → Identity providers page.

**NOTE**

After creating the identity provider, synchronization usually completes within two minutes. You can log in to the cluster as the user after the HTPasswd identity provider becomes available.

- The single, administrative user is visible on the Access control → Cluster Roles and Access page. The administration group membership of the user is also displayed.

12.8.8. Additional resources

- Accessing a cluster
- Understanding the ROSA deployment workflow

12.9. DELETING ACCESS TO A ROSA CLUSTER

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

12.9.1. Revoking dedicated-admin access using the rosa CLI

You can revoke access for a dedicated-admin user if you are the user who created the cluster, the organization administrator user, or the super administrator user.

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 the dedicated-admin access of a user:
Enter the following command to verify that your user no longer has `dedicated-admin` access. The output does not list the revoked user.

```
$ oc get groups dedicated-admins
```

### 12.9.2. Revoking cluster-admin access using the `rosa` CLI

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. Enter the following command to revoke the `cluster-admin` access of a user:

```
$ rosa revoke user cluster-admins --user=myusername --cluster=mycluster
```

2. Enter the following command to verify that the user no longer has `cluster-admin` access. The output does not list the revoked user.

```
$ oc get groups cluster-admins
```

### 12.10. DELETING A ROSA CLUSTER

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

#### 12.10.1. Prerequisites

- If Red Hat OpenShift Service on AWS created a VPC, you must remove the following items from your cluster before you can successfully delete your cluster:
  - Network configurations, such as VPN configurations and VPC peering connections
  - Any additional services that were added to the VPC

If these configurations and services remain, the cluster does not delete properly.

#### 12.10.2. Deleting a cluster

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

#### 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:

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

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

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

## 12.11. COMMAND QUICK REFERENCE FOR CREATING CLUSTERS AND USERS

### 12.11.1. Command quick reference list

If you have already created your first cluster and users, this list can serve as a command quick reference list when creating additional clusters and users.

```
## Configures your AWS account and ensures everything is setup correctly
$ rosa init

## Starts the cluster creation process (~30-40minutes)
$ rosa create cluster --cluster-name=<cluster_name>

## Connect your IDP to your cluster
$ rosa create idp --cluster=<cluster_name> --interactive

## Promotes a user from your IDP to dedicated-admin level
$ rosa grant user dedicated-admin --user=<idp_user_name> --cluster=<cluster_name>

## Checks if your install is ready (look for State: Ready),
## and provides your Console URL to login to the web console.
$ rosa describe cluster --cluster=<cluster_name>
```

### 12.11.2. Additional resources

- [Understanding the ROSA deployment workflow](#)