Red Hat AMQ 2021.Q2

Configuring AMQ Interconnect sites using the CLI

For Use with AMQ Interconnect 2.0 TECHNOLOGY PREVIEW

Last Updated: 2021-08-19

Red Hat AMQ 2021.Q2 Configuring AMQ Interconnect sites using the CLI

For Use with AMQ Interconnect 2.0 TECHNOLOGY PREVIEW

Legal Notice

Abstract

This guide describes how to install, configure, and manage AMQ Interconnect sites to build a service network.

Table of Contents

PREFACE

Making open source more inclusive

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message .

IMPORTANT

AMQ Interconnect 2.0 Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production.

These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. For more information about the support scope of Red Hat Technology Preview features, see https://access.redhat.com/support/offerings/techpreview.

CHAPTER 1. USING THE AMQ INTERCONNECT CLI

Using the **skupper** command-line interface (CLI) allows you to create and manage AMQ Interconnect sites from the context of the current namespace.

A typical workflow is to create a site, link sites together, and expose services to the service network.

1.1. INSTALLING THE SKUPPER CLI

Installing the **skupper** command-line interface (CLI) provides a simple method to get started with AMQ Interconnect.

Procedure

1. Ensure your subscription has been activated and your system is registered.
2. Subscribe to the required repositories:

   Red Hat Enterprise Linux 7

   ```
   $ sudo subscription-manager repos --enable=interconnect-2-for-rhel-7-server-rpms
   ```

   Red Hat Enterprise Linux 8

   ```
   $ sudo subscription-manager repos --enable=interconnect-2-for-rhel-8-x86_64-rpms
   ```

3. Use the **yum** or **dnf** command to install the **skupper** package:

   ```
   $ sudo yum install skupper
   ```

4. Verify the installation.

   ```
   $ skupper version
   client version 0.7.0-redhat-interconnect-2.0.0
   ```

1.2. CREATING A SITE USING THE CLI

A service network consists of AMQ Interconnect sites. This section describes how to create a site using the default settings.

Prerequisites

- The **skupper** CLI is installed.
- You are logged into the cluster.
- The services you want to expose on the service network are in the active namespace.

Procedure

1. Create a default site:

```
$ skupper init
```

2. Check the site:

```
$ skupper status

Skupper is enabled for namespace "west" in interior mode. It is not connected to any other sites.
```

The default settings include:

- console - The Skupper console is provisioned with a single user. The password for the **admin** user is stored in the **skupper-console-users** secret. For more information on the console, see Monitoring AMQ Interconnect sites using the console .
- site name - The site name defaults to the namespace name, for example, **west**.

1.3. CUSTOM SITES

The default **skupper init** creates sites that satisfy typical requirements.

If you require a custom configuration, note the following options:

- Creating a site without a console:

```
$ skupper init --enable-console false
```

- Configuring console authentication. There are a number of **skupper** options regarding authentication for the console:

  --console-auth <authentication-mode>

  Set the authentication mode for the console:

  - **openshift** - Use OpenShift authentication, so that users who have permission to log into OpenShift and view the Project (namespace) can view the console.
  - **internal** - Use AMQ Interconnect authentication, see the **console-user** and **console-password** options.
  - **unsecured** - No authentication, anyone with the URL can view the console.

  --console-user <username>

  Username for the console user when authentication mode is set to **internal**. Defaults to **admin**.

  --console-password <password>

  Password for the console user when authentication mode is set to **internal**. If not specified, a random passwords is generated.

1.4. LINKING SITES

A service network consists of AMQ Interconnect sites. This section describes how to link sites to form a service network.

Linking two sites requires a single initial directional connection. However:

- Communication between the two sites is bidirectional, only the initial linking is directional.
- The choice of direction for linking is typically determined by accessibility. For example, if you are linking an OpenShift Dedicated cluster with a CodeReady Containers cluster, you must link from the CodeReady Containers cluster to the OpenShift Dedicated cluster because that route is accessible.

Procedure

1. Determine the direction of the link. If both clusters are publicly addressable, then the direction is not significant. If one of the clusters is addressable from the other cluster, perform step 2 below on the addressable cluster.
2. Generate a token on the cluster that you want to link to:

   ```
   $ skupper token create <filename>
   ```

   where **<filename>** is the name of a YAML file that is saved on your local filesystem.

   This file contains a key and the location of the site that created it.

   NOTE

   Access to this file provides access to the service network. Protect it appropriately.

   For more information about protecting access to the service network, see {TokensLink}.

3. Use a token on the cluster that you want to connect from:
   a. Create a link to the service network:

      ```
      $ skupper link create <filename> [-name <link-name>]
      ```

      where **<filename>** is the name of a YAML file generated from the **skupper token create** command and **<link-name>** is the name of the link.

   b. Check the link:

      ```
      $ skupper link status
      Connection for link1 not active
      ```

      In this example no <link-name> was specified, the name defaulted to **link1**.

4. If you want to delete a link:

   ```
   $ skupper link delete <link-name>
   ```

   where **<link-name>** is the name of the link specified during creation.

1.5. EXPOSING SERVICES ON THE SERVICE NETWORK FROM A NAMESPACE

After creating a service network, exposed services can communicate across that network.

The **skupper** CLI has two options for exposing services that already exist in a namespace:

- **expose** supports simple use cases, for example, a deployment with a single service. See Section 1.5.1, "Exposing simple services on the service network" for instructions.
- **service create** and **service bind** is a more flexible method of exposing services, for example, if you have multiple services for a deployment. See Section 1.5.2, "Exposing complex services on the service network" for instructions.

1.5.1. Exposing simple services on the service network

This section describes how services can be enabled for a service network for simple use cases.

Procedure

1. Create a deployment, some pods, or a service in one of your sites, for example to create the backend service for the tutorial:

   ```
   $ kubectl create deployment hello-world-backend --image quay.io/skupper/hello-world-backend
   ```

   This step is not AMQ Interconnect-specific, that is, this process is unchanged from standard processes for your cluster.

2. Create a service that can communicate on the service network:

   ```
   $ skupper expose [deployment <name>|pods <selector>|statefulset <statefulsetname>|service <name>]
   ```

   where

   - **<name>** is the name of your deployment
   - **<selector>** is a pod selector
   - **<statefulsetname>** is the name of a statefulset

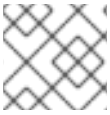
   For the example deployment in step 1, you create a service using the following command:

   ```
   $ skupper expose deployment/hello-world-backend --port 8080
   ```

   Options for this command include:

   --port <port-number>
   Specify the port number that this service is available on the service network.

   --target-port <port-number>
   Specify the port number of pods that you want to expose.



NOTE

If you do not specify ports, **skupper** uses the **containerPort** value of the deployment.

1.5.2. Exposing complex services on the service network

This section describes how services can be enabled for a service network for more complex use cases.

Procedure

1. Create a deployment, some pods, or a service in one of your sites, for example to create the backend service for the tutorial:

```
$ kubectl create deployment hello-world-backend --image quay.io/skupper/hello-world-backend
```

   This step is not AMQ Interconnect-specific, that is, this process is unchanged from standard processes for your cluster.

2. Create a service that can communicate on the service network:

```
$ skupper service create <name> <port>
```

   where

   - **<name>** is the name of the service you want to create
   - **<port>** is the port the service uses

   For the example deployment in step 1, you create a service using the following command:

```
$ skupper service create hello-world-backend 8080
```

3. Bind the service to a cluster service:

```
$ skupper service bind <service-name> <target-type> <target-name>
```

   where

   - **<service-name>** is the name of the service on the service network
   - **<target-type>** is the object you want to expose, **deployment**, **statefulset**, **pods**, or **service**.
   - **<target-name>** is the name of the cluster service

   For the example deployment in step 1, you bind the service using the following command:

```
$ skupper service bind hello-world-backend deployment hello-world-backend
```

1.6. EXPOSING SERVICES ON THE SERVICE NETWORK FROM A LOCAL MACHINE

After creating a service network, you can expose services from a local machine on the service network.

For example, if you run a database on a server in your data center, you can deploy a front end in a cluster that can access the data as if the database was running in the cluster.

1.6.1. Exposing simple local services to the service network

This section shows how to expose a single service running locally on a service network.

Prerequisites

- A service network. Only one site is required.
- Access to the service network.
- **qdrouterd**

Procedure

1. Run your service locally.
2. Log into your cluster and change to the namespace for your site.
3. Expose the service on the service network:

```
$ skupper gateway expose <service> localhost <port>
```

   - <service> - the name of the service on the service network.
   - <port> - the port that runs the service locally.

NOTE

You can also expose services from other machines on your local network, for example if MySQL is running on a dedicated server (with an IP address of **192.168.1.200**), but you are accessing the cluster from a machine in the same network:

```
$ skupper gateway expose mysql 192.168.1.200 3306
```

4. Check the status of Skupper gateways:

```
$ skupper gateway status
Gateway Definitions Summary

NAME    BINDS  FORWARDS  URL
<machine-name>  1      0        amqp://127.0.0.1:5672
```

   This shows that there is only one exposed service and that service is only exposing a single port (BIND). There are no ports forwarded to the local host.

   The URL field shows the underlying communication and can be ignored.

1.6.2. Working with complex local services on the service network

This section shows more advanced usage of skupper gateway.

1. Create a Skupper gateway:

```
$ skupper gateway init
```

2. Create a service that can communicate on the service network:

```
$ skupper service create <name> <port>
```

where

- **<name>** is the name of the service you want to create
- **<port>** is the port the service uses

For example:

```
$ skupper service create mydb 3306
```

3. Bind the service on the service network:

```
$ skupper gateway bind <service> <host> <port>
```

- <service> - the name of the service on the service network, **mydb** in the example above.
- <host> - the host that runs the service.
- <port> - the port the service is running on, **3306** from the example above.

4. Check the status of Skupper gateways:

```
$ skupper gateway status
Gateway Definitions Summary

NAME    BINDS  FORWARDS  URL
<machine-name>  1      0        amqp://127.0.0.1:5672
```

This shows that there is only one exposed service and that service is only exposing a single port (BIND). There are no ports forwarded to the local host.

The URL field shows the underlying communication and can be ignored.

You can create more services in the service network and bind more local services to expose those services on the service network.

5. Forward a service from the service network to the local machine.

```
$ skupper gateway forward <service> <port>
```

where

- **<service>** is the name of an existing service on the service network.
- **<port>** is the port on the local machine that you want to use.

1.7. CLI OPTIONS FOR WORKING WITH DIFFERENT CLUSTERS

By default, all **skupper** commands apply to the cluster you are logged into and the current namespace. The following **skupper** options allow you to override that behavior and apply to all commands:

--namespace <namespace-name>

Apply command to **<namespace-name>**. For example, if you are currently working on **frontend** namespace and want to initialize a site in the **backend** namespace:

```
$ skupper init --namespace backend
```

--kubeconfig <kubeconfig-path>

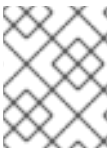
Path to the kubeconfig file - This allows you run multiple sessions to a cluster from the same client. An alternative is to set the **KUBECONFIG** environment variable. See the tutorial for an example of using kubeconfig files.

--context <context-name>

The kubeconfig file can contain defined contexts, and this option allows you to use those contexts.

CHAPTER 2. USING SKUPPER TOKENS

Skupper tokens allow you to create links between sites. You create a token on one site and use that token from the other site to create a link between the two sites.



NOTE

Although the linking process is directional, a Skupper link allows communication in both directions.

If both sites are equally accessible, for example, two public clouds, then it is not important where you create the token. However, when using a token, the site you link to must be accessible from the site you link from. For example, if you are creating a service network using both a public and private cluster, you must create the token on the public cluster and use the token from the private cluster.

There are two types of Skupper token:

Claim token (default)

A claim token can be restricted by:

- time - prevents token reuse after a specified period.
- usage - prevents creating multiple links from a single token.

All inter-site traffic is protected by mutual TLS using a private, dedicated certificate authority (CA). A claim token is not a certificate, but is securely exchanged for a certificate during the linking process. By implementing appropriate restrictions (for example, creating a single-use claim token), you can avoid the accidental exposure of certificates.

Cert token

You can use a cert token to create a link to the site which issued that token, it includes a valid certificate from that site.
All inter-site traffic is protected by mutual TLS using a private, dedicated certificate authority (CA). A cert token is a certificate issued by the dedicated CA. Protect it appropriately.

2.1. CREATING CLAIM TOKENS

You can use a claim token to create a link to the site which issued that token. It does not includes a certificate from that site, but a certificate is passed from the site when the claim token is used. A claim token can be restricted by time or usage.

Procedure

1. Log into the cluster.
2. Change to the namespace associated with the site.
3. Create a claim token, for example:

```
$ skupper token create $HOME/secret.yaml --expiry 30m0s --uses 2 -t claim
```

NOTE

Claim tokens are the default, the **-t claim** section of the command is unnecessary.

--expiry

The amount of time the token is valid in minutes and seconds, default **15m0s**.

--uses

The number of times you can use the token to create a link, default **1**.

Additional information

- See the Configuring AMQ Interconnect sites using the CLI for information on using the token to create links.

2.2. CREATING CERT TOKENS

A cert token allows you create many links to the service network from different sites without restrictions.

Procedure

1. Log into the cluster.
2. Change to the namespace associated with the site.
3. Create a cert token:

```
$ skupper token create $HOME/secret.yaml -t cert
```

NOTE

Cert tokens are always valid and can be reused indefinitely unless revoked as described in Section 2.3, "Revoking access to a site"

Additional information

- See the Configuring AMQ Interconnect sites using the CLI for information on using the token to create links.

2.3. REVOKING ACCESS TO A SITE

If a token is compromised, you can prevent unauthorized use of that token by invalidating all the tokens created from a site.

This option removes all links to the site and requires that you recreate any links to restore the service network.

1. Procedure
2. Log into the cluster.
3. Change to the namespace associated with the site.

4. Check the status of the site:

```
$ skupper status
Skupper is enabled for namespace "west" in interior mode. It is linked to 2 other sites.
```

5. Check outgoing links from the site:

```
$ skupper link status
Link link1 is active
```

   In this case, there are two links, and one outgoing link, meaning there is one incoming link.

6. Revoke access to the site from incoming links:

```
$ skupper revoke-access
```

7. Check the status of the site to see the revocation of access:

```
$ skupper status
Skupper is enabled for namespace "west" in interior mode. It is linked to 1 other site.
$ skupper link status
Link link1 is active
```

   The output shows that the **skupper revoke-access** command has revoked the incoming links, but outgoing links are still active.

   You can remove that link using the **skupper link delete link1** command, but to revoke access, you must follow this procedure while logged into the appropriate cluster.

Revised on 2021-08-19 12:45:07 UTC