Red Hat AMQ 2021.Q1

在 OpenShift 上部署 AMQ Interconnect

使用 AMQ Interconnect 1.10

Last Updated: 2023-02-26

Red Hat AMQ 2021.Q1 在 OpenShift 上部署 AMQ Interconnect

使用 AMQ Interconnect 1.10

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律通告

摘要

了解如何在 OpenShift Container Platform 上安装和部署 AMQ Interconnect。

目录

第 1 章 在 OPENSHIFT CONTAINER PLATFORM 中使用 AMQ INTERCONNECT

AMQ Interconnect 是一个轻量级 AMQP 1.0 消息路由器，用于构建用于混合云和 IoT/边缘计算部署的高弹性消息传递网络。AMQ Interconnect 会自动了解消息传递端点的地址（如客户端、服务器和消息代理），并在它们之间灵活地路由消息。

本文档论述了如何使用 AMQ Interconnect Operator 和它提供的 **Interconnect** 自定义资源定义 (CRD) 在 OpenShift Container Platform 上部署 AMQ 互联。CRD 定义了 AMQ Interconnect 部署，Operator 会在 OpenShift Container Platform 中创建和管理部署。

1.1. OPERATOR 是什么

Operator 是一种打包、部署和管理 Kubernetes 应用程序的方法。从概念上讲，Operator 会收集人类操作知识，并在软件中实现，以便更方便地与用户分享来实现一些常见的、复杂的任务。

在 OpenShift Container Platform 4.0 中，*Operator Lifecycle Manager*（OLM）可帮助用户安装、更新和管理所有 Operator 以及在用户集群中运行的关联服务的生命周期。Operator Lifecycle Manager 是 Operator Framework 的一部分，后者是一个开源工具包，用于以有效、自动化且可扩展的方式管理 Kubernetes 原生应用程序（Operator）。

OLM 默认在 OpenShift Container Platform 4.0 中运行，辅助集群管理员对集群上运行的 Operator 进行安装、升级和授予访问权。OpenShift Container Platform Web 控制台提供一些管理界面，供集群管理员安装 Operator，以及为特定项目授权以便使用集群上的可用 Operator 目录。

OperatorHub 是 OpenShift Container Platform 集群管理员用于发现、安装和升级 Operator 的图形界面。只需点一个按钮，即可从 OperatorHub 拉取并在 OperatorHub 中安装，并由 OLM 管理，为工程团队在开发、测试和生产环境中自助管理软件。

其他资源

- 如需有关 Operator 的更多信息，请参阅 OpenShift 文档。

1.2. 提供的自定义资源

AMQ Interconnect Operator 提供了 **Interconnect** 自定义资源定义 (CRD)，可让您像其他 OpenShift Container Platform API 对象一样与在 OpenShift Container Platform 上运行的 AMQ 互联部署进行交互。

Interconnect CRD 代表 AMQ Interconnect 路由器的部署。CRD 提供了在 OpenShift Container Platform 中定义路由器部署的很多不同方面，例如：

- AMQ Interconnect 路由器的数量
- 部署拓扑
- 连接性
- 地址语义

第 2 章 准备在 OPENSHIFT CONTAINER PLATFORM 上部署 AMQ INTERCONNECT

在 OpenShift Container Platform 上部署 AMQ Interconnect 之前，请执行以下步骤之一：

- 第 2.2 节 “添加 Red Hat Integration - AMQ Certificate Manager Operator”
- 第 2.1 节 “为 SSL/TLS 验证创建 secret”

如果要评估 AMQ Interconnect，您可以跳过这些步骤，但红帽建议您始终保护 AMQ Interconnect 通信。

2.1. 为 SSL/TLS 验证创建 SECRET

注意

如果您安装了 Red Hat Integration - AMQ Certificate Manager Operator，您可以跳过这个过程，使用 AMQ Certificate Manager 保护网络的说明将包括在相关流程中。OpenShift 使用名为 **Secret** 的对象来保存敏感信息，如 SSL/TLS 证书。如果要保护路由器间的流量、客户端流量或两者，那么您必须创建 SSL/TLS 证书和私钥，并将它们提供给 OpenShift 作为 secret。

流程

1. 如果您没有用于 inter-router 连接的现有证书颁发机构 (CA) 证书，请创建一个。
   这些命令为路由器连接创建一个自签名 CA 证书：

```
# Create a new directory for the inter-router certificates.
$ mkdir internal-certs

# Create a private key for the CA.
$ openssl genrsa -out internal-certs/ca-key.pem 2048

# Create a certificate signing request for the CA.
$ openssl req -new -batch -key internal-certs/ca-key.pem -out internal-certs/ca-csr.pem

# Self sign the CA certificate.
$ openssl x509 -req -in internal-certs/ca-csr.pem -signkey internal-certs/ca-key.pem -out
internal-certs/ca.crt
```

2. 为 CA 签名的路由器创建证书。
   这些命令创建私钥和证书，并使用上一步中创建的 CA 为证书签名：

```
# Create a private key.
$ openssl genrsa -out internal-certs/tls.key 2048

# Create a certificate signing request for the router.
$ openssl req -new -batch -subj "/CN=amq-interconnect.<project_name>.svc.cluster.local" -
key internal-certs/tls.key -out internal-certs/server-csr.pem

# Sign the certificate using the CA.
$ openssl x509 -req -in internal-certs/server-csr.pem -CA internal-certs/ca.crt -CAkey
internal-certs/ca-key.pem -out internal-certs/tls.crt -CAcreateserial
```

其中 **<project_name>** 是当前 OpenShift 项目的名称。

3. 创建包含私钥、路由器证书和 CA 证书的 secret。
此命令使用在前面的步骤中创建的密钥和证书创建 secret：

```
$ oc create secret generic inter-router-certs-secret --from-file=tls.crt=internal-certs/tls.crt  --from-file=tls.key=internal-certs/tls.key  --from-file=ca.crt=internal-certs/ca.crt
```

4. 如果要使用 SSL/TLS 验证客户端连接（与使用 SASL 验证客户端相比），为客户端连接创建一个 CA 证书。
这些命令为客户端连接创建自签名 CA 证书：

```
# Create a new directory for the client certificates.
$ mkdir client-certs

# Create a private key for the CA.
$ openssl genrsa -out client-certs/ca-key.pem 2048

# Create a certificate signing request for the CA.
$ openssl req -new -batch -key client-certs/ca-key.pem -out client-certs/ca-csr.pem

# Self sign the certificate.
$ openssl x509 -req -in client-certs/ca-csr.pem -signkey client-certs/ca-key.pem -out client-certs/ca.crt
```

5. 为 CA 签名的客户端连接创建证书。
这些命令创建私钥和证书，然后使用上一步中创建的 CA 为证书签名：

```
# Create a private key.
$ openssl genrsa -out client-certs/tls.key 2048

# Create a certificate signing request for the client connections
$ openssl req -new -batch -subj "/CN=<client_name>" -key client-certs/tls.key -out client-certs/client-csr.pem

# Sign the certificate using the CA.
$ openssl x509 -req -in client-certs/client-csr.pem -CA client-certs/ca.crt -CAkey client-certs/ca-key.pem -out client-certs/tls.crt -CAcreateserial
```

其中 **<client_name** > 对于每个路由器客户端是唯一的。

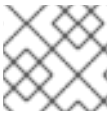
6. 创建包含使用在前面的步骤中创建的证书为客户端证书签名的 CA 证书的 secret：

```
$ oc create secret generic client-ca-secret --from-file=ca.crt=client-certs/ca.crt --from-file=tls.crt=client-certs/ca.crt --from-file=tls.key=client-certs/ca-key.pem
```

2.2. 添加 RED HAT INTEGRATION – AMQ CERTIFICATE MANAGER OPERATOR

Red Hat Integration - AMQ Certificate Manager Operator(cert-manager)是一个可选的 Kubernetes 附加组件，它解决了和管理 TLS 证书。Red Hat Integration - AMQ Interconnect Operator 使用它自动创建保护路由器网络所需的 TLS 证书。

您可以使用 *OperatorHub* 将 Operator 添加到 OpenShift Container Platform 集群。



注意

安装 Operator 需要 OpenShift 集群的管理员级权限。

Red Hat Integration - AMQ Certificate Manager Operator 不支持在 OpenShift Container Platform 4.9 或更高版本中。另外，您可以创建和管理 TLS 证书，如 第 2.1 节 “为 SSL/TLS 验证创建 secret” 所述。

安装之后，Operator 可供集群中的所有用户和项目使用。

先决条件

- 使用 **cluster-admin** 帐户访问 OpenShift Container Platform 4.6、4.7、4.8、4.9 或 4.10 集群。

流程

1. 在 OpenShift Container Platform Web 控制台中导航至 **Operators → OperatorHub**。
2. 从可用的 Operator 列表中选择 **Red Hat Integration - AMQ Certificate Manager Operator**，然后点 **Install**。
3. 在 **Operator 安装** 页面中，选择 **All namespaces on the cluster(default)**，然后点 **Install**。**Installed Operators** 页会显示 Operator 安装的状态。
4. 验证 Red Hat Integration - AMQ Certificate Manager Operator 已被显示，然后等待 **Status** 变为 **Succeeded**。
5. 如果安装不成功，请排除错误：
    a. 点 **Installed Operators** 页面中的 **Red Hat Integration - AMQ Certificate Manager Operator**。
    b. 选择 **Subscription** 选项卡，并查看任何故障或错误。

其他资源

- 有关 **cert-manager** 的更多信息，请参阅 cert-manager 文档。

第 3 章 添加 RED HAT INTEGRATION – AMQ INTERCONNECT OPERATOR

Red Hat Integration – AMQ Interconnect Operator 在 OpenShift Container Platform 中创建和管理 AMQ Interconnect 路由器网络。此 Operator 必须为每个使用它的项目单独安装。

安装 Operator 的选项有：

- 第 3.1 节 “使用 CLI 安装 Operator”
- 第 3.2 节 “使用 Operator Lifecycle Manager 安装 Operator”

注意

安装 Operator 需要 OpenShift 集群的管理员级权限。

3.1. 使用 CLI 安装 OPERATOR

本节中的步骤演示了如何使用 OpenShift 命令行界面(CLI)在给定的 OpenShift 项目中安装和部署最新版本的 Red Hat Integration – AMQ Interconnect Operator。

3.1.1. 获取 Operator 代码

此流程演示了如何访问和准备为 AMQ Interconnect 1.10 安装 Operator 最新版本所需的代码。

流程

1. 在网页浏览器中，导航到 AMQ Interconnect 版本的 **Software Downloads** 页面。
2. 确保 **Version** 下拉列表的值设置为 **1.10.7**，并且选择 **Releases** 选项卡。
3. 在 **AMQ Interconnect 1.10.7 Operator 安装和示例文件** 旁边，点 **Download**。
   下载 **amq-interconnect-operator-1.10.7-ocp-install-examples.zip** 压缩存档会自动开始。
4. 下载完成后，将存档移至您选择的安装目录。以下示例将存档 移到名为 **~/router/operator** 的目录。

```
$ mkdir ~/router
$ mv amq-interconnect-operator-1.10.7-ocp-install-examples.zip ~/router
```

5. 在您选择的安装目录中，提取存档的内容。例如：

```
$ cd ~/router
$ unzip amq-interconnect-operator-1.10.7-ocp-install-examples.zip
```

6. 切换到提取存档时创建的目录。例如：

```
$ cd operator
```

7. 以集群管理员身份登录 OpenShift Container Platform。例如：

```
$ oc login -u system:admin
```

8. 指定要安装 Operator 的项目。您可以创建新项目或切换到现有项目。

   a. 创建一个新项目：

   ```
   $ oc new-project <project-name>
   ```

   b. 或者，切换到现有项目：

   ```
   $ oc project <project-name>
   ```

9. 创建要与 Operator 搭配使用的服务帐户。

   ```
   $ oc create -f deploy/service_account.yaml
   ```

10. 为 Operator 创建角色。

    ```
    $ oc create -f deploy/role.yaml
    ```

11. 为 Operator 创建角色绑定。角色绑定根据您指定的名称将之前创建的服务帐户绑定到 Operator 角色。

    ```
    $ oc create -f deploy/role_binding.yaml
    ```

在下面的流程中，您要在项目中部署 Operator。

3.1.2. 使用 CLI 部署 Operator

本节中的步骤演示了如何使用 OpenShift 命令行界面(CLI)在 OpenShift 项目中为 AMQ Interconnect 1.10 部署最新版本的 Operator。

先决条件

- 您必须已经为 Operator 部署准备了 OpenShift 项目。请参阅 第 3.1.1 节 “获取 Operator 代码”。
- 在按照本节中的步骤操作前，您必须首先完成 Red Hat Container Registry 身份验证 中描述的步骤。

流程

1. 在 OpenShift 命令行界面(CLI)中，作为集群管理员登录到 OpenShift Container Platform。例如：

   ```
   $ oc login -u system:admin
   ```

2. 切换到您之前为 Operator 部署准备的项目。例如：

   ```
   $ oc project <project-name>
   ```

3. 切换到之前提取 Operator 安装存档时创建的目录。例如：

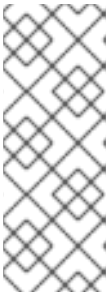
   ```
   $ cd ~/router/operator/qdr-operator-1.10-ocp-install-examples
   ```

4. 部署 Operator 中包含的 CRD。您必须在部署和启动 Operator 前，在 OpenShift 集群中安装 CRD。

```
$ oc create -f deploy/crds/interconnectedcloud_v1alpha1_interconnect_crd.yaml
```

5. 将与红帽生态系统目录中用于身份验证的帐户关联的 pull secret 与 OpenShift 项目 **的默认、部署器和 构建器服务帐户** 相关联。

```
$ oc secrets link --for=pull default <secret-name>
$ oc secrets link --for=pull deployer <secret-name>
$ oc secrets link --for=pull builder <secret-name>
```



注意

在 OpenShift Container Platform 4.1 或更高版本中，您还可以使用 Web 控制台将 pull secret 与您要在其中部署容器镜像的项目（如 AMQ Interconnect Operator）关联。为此，请单击 Administration → Service Accounts。指定与您在 Red Hat Container Registry 中用于身份验证的帐户关联的 pull secret。

6. **部署 Operator。**

```
$ oc create -f deploy/operator.yaml
```

7. 验证 **Operator 是否正在运行：**

```
$ oc get pods -l name=qdr-operator
```

如果输出没有报告 pod 正在运行，请使用以下命令来确定阻止它运行的问题：

```
$ oc describe pod -l name=qdr-operator
```

8. 验证 **CRD 是否已在集群中注册，并查看 CRD 详情：**

```
$ oc get crd
$ oc describe crd interconnects.interconnectedcloud.github.io
```

注意

建议您在给定的 OpenShift 项目中仅部署 AMQ Interconnect Operator 的单个实例。不建议将 Operator 部署的 replicas 元素设置为大于 1 的值，或者不要 在同一项目中部署 Operator。

其他资源

- 有关安装使用 OperatorHub 图形界面的 AMQ Interconnect Operator 的替代方法，请参阅 第 3.2 节 “使用 Operator Lifecycle Manager 安装 Operator”。

3.2. 使用 OPERATOR LIFECYCLE MANAGER 安装 OPERATOR

本节中的步骤演示了如何使用 OperatorHub 在给定的 OpenShift 项目中安装和部署最新版本的 Red Hat Integration - AMQ Interconnect Operator。

在 OpenShift Container Platform 4.1 及更高版本中，*Operator Lifecycle Manager* (OLM)可帮助用户安装、更新并通常管理所有 Operator 以及在用户集群中运行的关联服务的生命周期。Operator Lifecycle Manager 是 Operator Framework 的一部分，后者是一个开源工具包，用于以有效、自动化且可扩展的方式管理 Kubernetes 原生应用程序（Operator）。

先决条件

- 使用 cluster-admin 帐户访问 OpenShift Container Platform 4.6、4.7、4.8、4.9 或 4.10 集群。
- Red Hat Integration - AMQ Certificate Manager Operator 安装在 OpenShift Container Platform 集群中（如果需要）。

流程

1. 在 OpenShift Container Platform Web 控制台中导航至 Operators → OperatorHub。
2. 从可用的 Operator 列表中选择 Red Hat Integration - AMQ Interconnect Operator，然后点 Install。
3. 在 Operator 安装 页面中，选择要安装 Operator 的命名空间，然后点 Install。

Installed Operators 页会显示 **Operator** 安装的状态。

4. 验证 **AMQ Interconnect Operator** 已被显示，然后等待 **Status** 变为 **Succeeded**。

5. 如果安装不成功，请排除错误：

   a. 点 **Installed Operators** 页面中的 **Red Hat Integration - AMQ Interconnect Operator**。

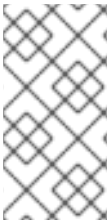
   b. 选择 **Subscription** 选项卡，并查看任何故障或错误。

第 4 章 安装 RED HAT INTEGRATION - AMQ INTERCONNECT OPERATOR

在没有或受限的互联网访问的环境中，安装 Red Hat Integration - AMQ Interconnect Operator，如 *第 3 章 添加 Red Hat Integration - AMQ Interconnect Operator* 所述。本节介绍如何在受限环境中安装 Red Hat Integration - AMQ Interconnect Operator，方法是将所需组件镜像到集群。

先决条件

- OpenShift Container Platform 集群版本 4.6、4.7、4.8、4.9 或 4.10
- 一个 RHEL 机器：
  - podman 1.9.3 或更高版本
  - OpenShift 文档中所述的 opm CLI
  - oc CLI 版本 4.9.9 或更高版本
- 网络访问
  - 网络访问 Red Hat Container Registry
  - 网络访问 OpenShift Container Platform 集群



注意

您只需要在镜像期间访问 Red Hat Container Registry。您不需要同时访问 Red Hat Container Registry 和 OpenShift Container Platform 集群。

以下部分描述了所需步骤：

- 第 4.1 节 “设置 OpenShift Container Platform 集群”
- 第 4.2 节 “在 RHEL 机器上创建 AMQ Interconnect 镜像”
- 第 4.3 节 “将镜像推送到 OpenShift Container Platform 集群”

4.1. 设置 OPENSHIFT CONTAINER PLATFORM 集群

在 OpenShift Container Platform 集群上完成以下步骤，为镜像(mirror)过程进行准备：

1. 以 cluster-admin 身份登录集群。
2. 使用 CLI 或 OpenShift 控制台禁用默认目录的源：
   a. 对于 CLI，为 OperatorHub 设置 disableAllDefaultSources: true ：

```
$ oc patch OperatorHub cluster --type json \
-p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

   b. 对于 OpenShift 控制台，导航到 Administration → Cluster Settings → Configuration → OperatorHub。

   在 OperatorHub 页面中，点击 Sources 选项卡，再禁用源。

4.2. 在 RHEL 机器上创建 AMQ INTERCONNECT 镜像

在 RHEL 计算机上完成以下步骤，为镜像(mirror)过程进行准备：

先决条件

- 访问 registry.redhat.io

1. 从 RHEL 机器登录到 registry.redhat.io。

```
$ podman login -u USERNAME -p PASSWORD registry.redhat.io
```

2. 只在 Operator 列表中保留 Interconnect Operator:

```
$ opm index prune -f registry.redhat.io/redhat/redhat-operator-index:v<openshift-version> -p amq7-interconnect-operator -t <cluster-domain>:<registry-port>/iib:my-operator-iib
```

其中

- <OpenShift-version> 是 OpenShift Container Platform 的版本，如 4.9。
- <cluster-domain> 是 OpenShift Container Platform 集群的域名，如 mycluster.example.com。
- <registry-port> 是 OpenShift Container Platform 集群中 registry 使用的端口号，默认为 5000。

验证您只创建了 Interconnect Operator 的 podman 镜像：

```
$ podman images | grep my-operator-iib <cluster-domain>:<registry-port>/iib

my-operator-iib  39b6148e6981  3 days ago     138 MB
```

4.3. 将镜像推送到 OPENSHIFT CONTAINER PLATFORM 集群

先决条件

- 从 RHEL 机器访问 OpenShift Container Platform 集群。

1. 从 RHEL 机器，将镜像推送到集群 registry：

```
$ podman push <cluster-domain>:<registry-port>/iib:my-operator-iib
```

2. 创建镜像过程所需的三个文件：

```
$ /usr/local/bin/oc adm catalog mirror  \
  <cluster-domain>:<registry-port>/iib:my-operator-iib \
  <cluster-domain>:<registry-port> \
  -a /home/customer-user/.docker/config.json \
  --insecure=true \
  --registry-config /home/customer-user/.docker/config.json \
  --index-filter-by-os=linux/amd64 \
  --manifests-only
```

3. 请确定有以下文件：

   - CatalogSource.yaml - 描述 catalogSource 的 YAML 文件。
   - ImageContentSourcePolicy.yaml - 使用红帽 registry 中的地址映射内部 registry 中的镜像的 YAML 文件。
   - mapping.txt- 将镜像镜像(mirror)到内部 registry 的文本文件。

4. 编辑 mapping.txt，以仅列出您要镜像的镜像。

   该文件的格式如下：

```
[ Operator address on RedHat registry : Operator SHA ] = [ Operator address on internal mirror registry : tag ]
```

   mapping.txt 文件示例：

```
registry.redhat.io/amq7/amq-interconnect@sha256:6101cc735e4d19cd67c6d80895c425ecf6f1d2604d88f999fa0cae57a7d6abaf=<cluster-domain>:<registry-port>/amq7-amq-interconnect:f793b0cc
registry.redhat.io/amq7/amq-interconnect-operator@sha256:8dd53290c909589590b88a1544d854b4ad9f8b4a639189597c0a59579bc60c40=<cluster-domain>:<registry-port>/amq7-amq-interconnect-operator:73c142ff
registry.redhat.io/amq7/amq-interconnect-operator-metadata@sha256:799ce48905d5d2a91b42e2a7943ce9b756aa9da80f6924be06b2a6275ac90214=<cluster-domain>:<registry-port>/amq7-amq-interconnect-operator-metadata:14cc4a4e
```

5. 镜像所需的镜像

```
$ /usr/local/bin/oc image mirror \
  -f mapping-ic.yaml \
  -a /home/customer-user/.docker/config.json \
  --insecure=true \
  --registry-config /home/customer-user/.docker/config.json \
  --keep-manifest-list=true
```

6. 配置 ImageContentSourcePolicy (ICSP)名称：

在文件 imageContentSourcePolicy.yaml 中设置字段 'name'，例如 my-operator-icsp

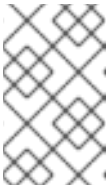
ICSP 片断示例：

```
---
apiVersion: operator.openshift.io/v1alpha1
kind: ImageContentSourcePolicy
metadata:
   labels:
      operators.openshift.org/catalog: "true"
   name: my-operator-icsp
spec:
   repositoryDigestMirrors:
   - mirrors:
      - <cluster-domain>:<registry-port>/amq7-amq-interconnect-operator
      source: registry.redhat.io/amq7/amq-interconnect-operator
```

7. 应用策略(ICSP)文件：

```
$ /usr/local/bin/oc create -f imageContentSourcePolicy.yaml
```

应用此文件后，所有集群节点都会自动重置。您可以使用 oc get nodes 或 OpenShift 控制台检查节点状态，方法是导航到 Compute → Nodes。



注意

在继续前，请确保所有节点都处于 Ready 状态。

8. 配置 catalogSource 名称：

   在 catalogSource.yaml 文件中设置 字段名称，如 my-operator-catalog

   catalogSource.yaml 文件示例：

```
apiVersion: operators.coreos.com/v1alpha1
kind: CatalogSource
metadata:
  name: iib
  namespace: openshift-marketplace
spec:
  image: <cluster-domain>:<registry-port>/iib:my-operator-iib
  sourceType: grpc
```

9. 应用目录源配置以完成 Red Hat Integration - AMQ Interconnect Operator 的安装：

```
$ /usr/local/bin/oc apply -f catalogSource.yaml
```

10. 如 所述，通过部署路由器确保安装正常工作。 第 5.1 节 “创建内路由器部署”

第 5 章 创建路由器网络

要创建 AMQ 互联路由器网络，您可以在 互联 自定义资源中定义部署，然后应用它。AMQ Interconnect Operator 通过调度必要的 Pod 并创建任何所需资源来创建部署。

本节中的步骤演示以下路由器网络拓扑：

- interior router mesh
- 带有边缘路由器的路由器网格以实现可扩展性
- 连接两个 OpenShift 集群的集群路由器网络

先决条件

- AMQ Interconnect Operator 已安装在 OpenShift Container Platform 项目中。

5.1. 创建内路由器部署

相互间的路由器建立相互连接，并自动计算网络中的最低成本路径。

流程

此流程创建三个路由器的内路由器网络。路由器在网格拓扑中相互自动连接，它们的连接可以通过 mutual SSL/TLS 身份验证进行保护。

1. 创建描述 interior 路由器部署的 Interconnect 自定义资源 YAML 文件。

router-mesh.yaml 文件示例

```
apiVersion: interconnectedcloud.github.io/v1alpha1
kind: Interconnect
metadata:
  name: router-mesh
```

```
spec:
 deploymentPlan:
   role: interior (1)
   size: 3 (2)
   placement:  Any (3)
```

(1)

部署中路由器的操作模式。Operator 将在网格拓扑中自动连接间路由器。

(2)

要创建的路由器数量。

(3)

每个路由器在单独的 Pod 中运行。放置(Placement)定义了 Operator 应该调度并放置 Pod 的位置。您可以选择以下放置选项：

任意

Pod 可以在 OpenShift Container Platform 集群的任何节点上运行。

每

Operator 将路由器 Pod 放置到集群中的每个节点上。如果选择这个选项，则不需要 Size 属性，路由器数量与集群中的节点数量对应。

反关联性

Operator 确保多个路由器 Pod 不在集群的同一节点上运行。如果大小超过集群中的节点数量，则无法调度的额外 Pod 将保持 Pending 状态。

2. 创建 YAML 文件中描述的路由器部署。

```
$ oc apply -f router-mesh.yaml
```

Operator 在网格拓扑中创建部署交集，该路由器使用默认的地址语义。它还会创建一个服务，在其中可访问该路由器，以及您可以访问 Web 控制台的路由。

3. 验证路由器网格是否已创建且 Pod 是否正在运行。

每个路由器在单独的 Pod 中运行。它们使用 Operator 创建的 Service 会自动连接。

```
$ oc get pods
NAME                                  READY   STATUS    RESTARTS   AGE
interconnect-operator-587f94784b-4bzdx   1/1     Running   0          52m
router-mesh-6b48f89bd-588r5              1/1     Running   0          40m
router-mesh-6b48f89bd-bdjc4              1/1     Running   0          40m
router-mesh-6b48f89bd-h6d5r              1/1     Running   0          40m
```

4. 检查路由器部署。

```
$ oc get interconnect/router-mesh -o yaml
apiVersion: interconnectedcloud.github.io/v1alpha1
kind: Interconnect
...
spec:
  addresses:  1
  - distribution: closest
    prefix: closest
  - distribution: multicast
    prefix: multicast
  - distribution: closest
    prefix: unicast
  - distribution: closest
    prefix: exclusive
  - distribution: multicast
    prefix: broadcast
  deploymentPlan:  2
    livenessPort: 8888
    placement: Any
    resources: {}
    role: interior
    size: 3
  edgeListeners:  3
  - port: 45672
  interRouterListeners:  4
  - authenticatePeer: true
    expose: true
    port: 55671
    saslMechanisms: EXTERNAL
    sslProfile: inter-router
  listeners:  5
  - port: 5672
  - authenticatePeer: true
    expose: true
    http: true
    port: 8080
  - port: 5671
    sslProfile: default
  sslProfiles:  6
  - credentials: router-mesh-default-tls
```

```
  name: default
- caCert: router-mesh-inter-router-tls
  credentials: router-mesh-inter-router-tls
  mutualAuth: true
  name: inter-router
users: router-mesh-users (7)
```

(1)

默认地址配置。发送到与这些前缀不匹配的地址的所有消息都会在 均衡的 任意广播模式中分发。

(2)

部署了三个间路由器的路由器网格。

(3)

每个相互路由器侦听端口 45672，以获取来自边缘路由器的连接。

(4)

在端口 55671 中，相互路由器互相连接。这些路由器连接通过 SSL/TLS mutual 验证进行保护。inter-router SSL Profile 包含 Operator 生成的证书详情。

(5)

每个 interior 路由器侦听来自以下端口的外部客户端的连接：

- 5672 - 从消息传递应用程序取消安全连接。
- 5671 - 从消息传递应用程序的安全连接。
- 8080 - AMQ Interconnect Web 控制台访问。应用默认用户名/密码安全性。

(6)

使用 Red Hat Integration - AMQ Certificate Manager Operator，Red Hat Integration - AMQ Interconnect Operator 会自动创建两个 SSL 配置集：

- inter-router - Operator 通过创建证书颁发机构(CA)并为每个间路由器生成 CA 签名的证书，从而保护路由器的 inter-router 网络与 mutual TLS 身份验证的安全。

- Default - Operator 为消息应用程序创建 TLS 证书，以连接到端口 5671 上的 interior 路由器。

7 AMQ Interconnect web 控制台使用用户名/密码身份验证进行保护。Operator 会自动生成凭证并将其存储在 router-mesh-users Secret 中。

5.2. 创建边缘路由器部署

您可以通过添加边缘路由器部署来高效地扩展路由器网络。边缘路由器充当消息传递应用程序的连接原则。每个边缘路由器维护一个到交集的 uplink 连接，而消息应用程序连接至边缘路由器，以发送和接收消息。

先决条件

- 部署了相互路由器网格。更多信息请参阅 第 5.1 节 “创建内路由器部署”。

流程

此流程在每个 OpenShift Container Platform 集群的节点上创建一个边缘路由器，并将其连接到之前创建的 interior 路由器网格。

1. 创建描述边缘路由器部署的 Interconnect 自定义资源 YAML 文件。

edge-routers.yaml 文件示例

```
apiVersion: interconnectedcloud.github.io/v1alpha1
kind: Interconnect
metadata:
  name: edge-routers
spec:
  deploymentPlan:
    role: edge
    placement: Every 1
  edgeConnectors: 2
    - host: router-mesh 3
      port: 45672 4
```

1. 边缘路由器 Pod 将部署到 OpenShift Container Platform 集群的每个节点上。此放置有助于平衡集群中的消息传递应用程序流量。Operator 将创建一个 DaemonSet，以确保调度的 pod 数量始终与集群中的节点数量对应。

2. 边缘连接器定义从边缘路由器到内路由器的连接。

3. 为 interior 路由器创建的 Service 的名称。

4. 路由器在边缘连接中侦听的端口。默认值为 45672。

2. 创建 YAML 文件中描述的边缘路由器：

```
$ oc apply -f edge-routers.yaml
```

Operator 在 OpenShift Container Platform 集群的每个节点上部署一个边缘路由器，并将它们连接到 interior 路由器。

3. 验证边缘路由器是否已创建且 Pod 是否正在运行。

每个路由器在单独的 Pod 中运行。每个边缘路由器都连接到之前创建的任意路由器。

```
$ oc get pods
NAME                                    READY   STATUS    RESTARTS   AGE
edge-routers-2jz5j                      1/1     Running   0          33s
edge-routers-fhlxv                      1/1     Running   0          33s
edge-routers-gg2qb                      1/1     Running   0          33s
edge-routers-hj72t                      1/1     Running   0          33s
interconnect-operator-587f94784b-4bzdx  1/1     Running   0          54m
router-mesh-6b48f89bd-588r5             1/1     Running   0          42m
router-mesh-6b48f89bd-bdjc4             1/1     Running   0          42m
router-mesh-6b48f89bd-h6d5r             1/1     Running   0          42m
```

5.3. 创建集群间路由器网络

根据您是否在使用 AMQ 证书管理器，都有不同的步骤来创建集群间路由器网络。

- 第 5.3.2 节 “使用 AMQ 证书管理器创建集群间路由器网络”
- 第 5.3.1 节 “使用证书颁发机构创建集群间路由器网络”

5.3.1. 使用证书颁发机构创建集群间路由器网络

您可以从在不同 OpenShift Container Platform 集群中运行的路由器创建路由器网络。这可让您连接在独立集群中运行的应用程序。

先决条件

- 您已创建 secret，为每个路由器定义现有证书。

流程

此流程以两个不同的 OpenShift Container Platform 集群（cluster1 和 cluster2）创建路由器部署，并将它们连接在一起，形成集群间路由器网络。路由器部署之间的连接通过 SSL/TLS mutual 验证进行保护。

1. 在第一个 OpenShift Container Platform 集群中(cluster1)中，创建一个描述 interior 路由器部署的 Interconnect 自定义资源 YAML 文件。

   这个示例创建带有默认配置的单个 interior 路由器。

   cluster1-router-mesh.yaml 文件示例

```
apiVersion: interconnectedcloud.github.io/v1alpha1
kind: Interconnect
metadata:
  name: cluster1-router-mesh
spec:
  interRouterListeners:
    - authenticatePeer: true (1)
```

```
    host: 0.0.0.0 2
    port: 55672 3
    saslMechanisms: EXTERNAL 4
    sslProfile: inter-router-profile 5
    expose: true 6
sslProfiles:
 - caCert: inter-router-certs-secret 7
    credentials: inter-router-certs-secret 8
    name: inter-router-profile 9
```

1

authenticatePeer 必须设为 true 才能使用 TLS 证书进行身份验证

2

侦听器的主机

3

侦听器端口

4

SASL 机制进行身份验证，对 TLS 证书使用 EXTERNAL

5

用于验证客户端的 SSL 配置档案名称

6

公开路由，以便能够从集群外部访问端口

7

包含 ca.crt 名称的集群 secret 或您的 CA（在这种情况下，您使用了凭证中使用的同一 secret，否则必须具有 tls.crt）

8

包含 tls.crt 和 tls.key 文件的 CA 证书的集群 secret 名称

9

用于 interRouterListener 的 SSL-profile 名称

2. 创建 YAML 文件中描述的路由器部署。

```
$ oc apply -f cluster1-router-mesh.yaml
```

Red Hat Integration - AMQ Interconnect Operator 创建一个带有默认配置的交集，以及一个监听程序来验证其他路由器。

3. 登录第二个 OpenShift Container Platform 集群(cluster2)，并切换到要创建第二个路由器部署的项目。

4. 在 cluster2 中，创建一个 互连 自定义资源 YAML 文件来描述路由器部署。

cluster2-router-mesh.yaml 文件示例

```
apiVersion: interconnectedcloud.github.io/v1alpha1
kind: Interconnect
metadata:
  name: cluster2-router-mesh
spec:
  sslProfiles:
  - name: inter-router-profile 1
    credentials: inter-router-certs-secret
    caCert: inter-router-certs-secret
  interRouterConnectors:
  - host: cluster1-router-mesh-port-55672-myproject.cluster1.openshift.com 2
    port: 443
    verifyHostname: false
    sslProfile: inter-router-profile
    name: cluster1
```

1

此 SSL 配置文件定义了连接到 cluster1 中路由器部署所需的证书。

2 cluster1 上 inter-router 侦听器的路由 URL。

5. 创建 YAML 文件中描述的路由器部署。

```
$ oc apply -f cluster2-router-mesh.yaml
```

6. 验证路由器已连接。

这个示例显示 cluster2 中路由器到 cluster1 中的路由器的连接。

```
$ oc exec cluster2-fb6bc5797-crvb6 -it -- qdstat -c
Connections
  id    host                                                container                role         dir
security                        authentication  tenant

========================================================================================
========================================================================================
================================================
  1     cluster1-router-mesh-port-55672-myproject.cluster1.openshift.com:443  cluster1-router-
mesh-54cffd9967-9h4vq  inter-router  out  TLSv1/SSLv3(DHE-RSA-AES256-GCM-SHA384)
x.509
```

5.3.2. 使用 AMQ 证书管理器创建集群间路由器网络

您可以从在不同 OpenShift Container Platform 集群中运行的路由器创建路由器网络。这可让您连接在独立集群中运行的应用程序。

流程

此流程以两个不同的 OpenShift Container Platform 集群（cluster1 和 cluster2）创建路由器部署，并将它们连接在一起，形成集群间路由器网络。路由器部署之间的连接通过 SSL/TLS mutual 验证进行保护。

1. 在第一个 OpenShift Container Platform 集群中(cluster1)中，创建一个描述 interior 路由器部署的 Interconnect 自定义资源 YAML 文件。

这个示例创建带有默认配置的单个 interior 路由器。

cluster1-router-mesh.yaml 文件示例

```
apiVersion: interconnectedcloud.github.io/v1alpha1
kind: Interconnect
metadata:
  name: cluster1-router-mesh
spec: {}
```

2. 创建 YAML 文件中描述的路由器部署。

```
$ oc apply -f cluster1-router-mesh.yaml
```

Red Hat Integration - AMQ Interconnect Operator 创建一个带有默认配置的 interior 路由器。它使用 Red Hat Integration - AMQ Certificate Manager Operator 创建证书颁发机构(CA)并生成由 CA 签名的证书。

3. 在第二个 OpenShift Container Platform 集群中为路由器部署生成额外证书(cluster2)。

cluster2 中的路由器部署需要 cluster1 CA 发布的证书。

a. 创建 证书 自定义资源 YAML 文件以请求证书。

certificate-request.yaml 文件示例

```
apiVersion: certmanager.k8s.io/v1alpha1
kind: Certificate
metadata:
  name: cluster2-inter-router-tls
spec:
  commonName: cluster1-router-mesh-myproject.cluster2.openshift.com
  issuerRef:
```

```
    name: cluster1-router-mesh-inter-router-ca (1)
  secretName: cluster2-inter-router-tls-secret
---
```

(1) 为 **cluster1** 创建 **inter-router CA** 的发行者名称。默认情况下，**Issuer** 的名称为 **<** ***application-name*****>-inter-router-ca**。

b. 创建 **YAML** 文件中描述的证书。

```
$ oc apply -f certificate-request.yaml
```

c. 提取您生成的证书。

```
$ mkdir /tmp/cluster2-inter-router-tls
$ oc extract secret/cluster2-inter-router-tls-secret --to=/tmp/cluster2-inter-router-tls
```

4. 登录第二个 **OpenShift Container Platform** 集群(**cluster2**)，并切换到要创建第二个路由器部署的项目。

5. 在 **cluster2** 中，创建一个包含您生成的证书的 **Secret**。

```
$ oc create secret generic cluster2-inter-router-tls-secret --from-file=/tmp/cluster2-inter-router-tls
```

6. 在 **cluster2** 中，创建一个 互连 自定义资源 **YAML** 文件来描述路由器部署。

cluster2-router-mesh.yaml 文件示例

```
apiVersion: interconnectedcloud.github.io/v1alpha1
kind: Interconnect
metadata:
  name: cluster2-router-mesh
spec:
  sslProfiles:
```

```
- name: inter-cluster-tls (1)
  credentials: cluster2-inter-router-tls-secret
  caCert: cluster2-inter-router-tls-secret
interRouterConnectors:
- host: cluster1-router-mesh-port-55671-myproject.cluster1.openshift.com (2)
  port: 443
  verifyHostname: false
  sslProfile: inter-cluster-tls
```

(1) 此 SSL 配置文件定义了连接到 cluster1 中路由器部署所需的证书。

(2) cluster1 上 inter-router 侦听器的路由 URL。

7. 创建 YAML 文件中描述的路由器部署。

```
$ oc apply -f cluster2-router-mesh.yaml
```

8. 验证路由器已连接。

这个示例显示 cluster2 中路由器到 cluster1 中的路由器的连接。

```
$ oc exec cluster2-fb6bc5797-crvb6 -it -- qdstat -c
Connections
  id    host                                                container                role         dir
security                        authentication  tenant

================================================================================
================================================================================
=============================================
  1     cluster1-router-mesh-port-55671-myproject.cluster1.openshift.com:443  cluster1-router-
mesh-54cffd9967-9h4vq  inter-router  out  TLSv1/SSLv3(DHE-RSA-AES256-GCM-SHA384)
x.509
```

第 6 章 将客户端连接到路由器网络

在创建了路由器网络后，您可以将客户端（消息传递应用程序）连接到它，以便它们能够开始发送和接收消息。

默认情况下，Red Hat Integration - AMQ Interconnect Operator 为路由器部署创建一个服务，并为客户端访问配置以下端口：

- 5672 用于没有身份验证的普通 AMQP 流量
- 用于使用 TLS 验证保护的 AMQP 流量的 5671

要将客户端连接到路由器网络，您可以执行以下操作：

- 如果任何客户端都位于 OpenShift 集群之外，请公开端口，以便它们能够连接到路由器网络。
- 将您的客户端配置为连接到路由器网络。

6.1. 为 OPENSHIFT CONTAINER PLATFORM 外部的客户端公开端口

您可以公开端口，以便 OpenShift Container Platform 集群外部的客户端连接到路由器网络。

流程

1. 开始编辑 Interconnect 自定义资源 YAML 文件，该文件描述了您要公开端口的路由器部署。

```
$ oc edit -f router-mesh.yaml
```

2. 在 spec.listeners 部分中，公开您希望集群外的客户端的每个端口都可以访问。

   在本例中，会公开端口 5671。这可让集群外部的客户端与路由器网络进行身份验证并连接到路由器网络。

router-mesh.yaml 文件示例

```
apiVersion: interconnectedcloud.github.io/v1alpha1
kind: Interconnect
metadata:
  name: router-mesh
spec:
  ...
  listeners:
    - port: 5672
    - authenticatePeer: true
      expose: true
      http: true
      port: 8080
    - port: 5671
      sslProfile: default
      expose: true
  ...
```

Red Hat Integration - AMQ Interconnect Operator 会创建一个 Route，集群外部的客户端可以使用它连接到路由器网络。

6.2. 客户端连接身份验证

当您创建路由器部署时，Red Hat Integration - AMQ Interconnect Operator 使用 Red Hat Integration - AMQ Certificate Manager Operator 为客户端验证创建默认的 SSL/TLS 证书，并为 SSL 加密配置端口 5671。

6.3. 配置客户端以连接到路由器网络

您可以连接在与路由器网络、不同集群或 OpenShift 外部运行的 OpenShift 集群中运行的消息传递客户端，以便它们能够交换消息。

先决条件

- 如果客户端位于 OpenShift Container Platform 集群之外，则必须公开连接端口。更多信息请参阅 第 6.1 节 “为 OpenShift Container Platform 外部的客户端公开端口”。

流程

- 要将客户端连接到路由器网络，请使用以下连接 URL 格式：

```
<scheme>://[<username>@]<host>[:<port>]
```

<scheme>

使用以下之一：

- AMQP - 同一 OpenShift 集群中的未加密 TCP
- amqps - 用于使用 SSL/TLS 验证的安全连接
- amqpws - AMQP over WebSockets 用于来自 OpenShift 集群外部的未加密的连接

<username>

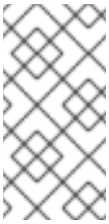
如果您使用用户名/密码身份验证部署了路由器网格，请提供客户端的用户名。

<host>

如果客户端与路由器网络位于同一个 OpenShift 集群中，请使用 OpenShift Service 主机名。否则，使用路由的主机名。

<port>

如果您要连接到 Route，您必须指定端口。要在不受保护的连接上连接，请使用端口 80。否则，要在安全连接中进行连接，请使用端口 443。



注意

要在不安全的连接上进行连接（端口 80），客户端必须通过 WebSockets(mqpws)使用 AMQP。

下表显示了一些示例连接 URL。

URL	描述
amqp://admin@router-mesh:5672	客户端和路由器网络都位于同一 OpenShift 集群中，因此服务主机名用于连接 URL。在这种情况下，会实施用户名/密码身份验证，这需要提供用户名 (**admin**)。
amqps://router-mesh-myproject.mycluster.com:443	客户端位于 OpenShift 外部，因此 Route 主机名用于连接 URL。在这种情况下，会实现 SSL/TLS 身份验证，这需要 **amqps** 方案和端口 **443**。
amqpws://router-mesh-myproject.mycluster.com:80	客户端位于 OpenShift 外部，因此 Route 主机名用于连接 URL。在本例中，没有实施身份验证，这意味着客户端必须使用 **amqpws** 方案和端口 **80**。

第 7 章 连接到外部服务

您可以将路由器连接到外部服务，如消息代理。服务可能与路由器网络在同一 OpenShift 集群中运行，或者在 OpenShift 外部运行。

先决条件

- 您必须有权访问消息代理。

流程

这个步骤描述了如何将路由器连接到代理并配置链路路由来连接消息传递客户端。

1. 开始编辑 Interconnect Custom Resource YAML 文件，该文件描述了您要连接到代理的路由器部署。

```
$ oc edit -f router-mesh.yaml
```

2. 在 spec 部分中，配置连接和链路路由。

router-mesh.yaml 文件示例

```
apiVersion: interconnectedcloud.github.io/v1alpha1
kind: Interconnect
metadata:
  name: router-mesh
spec:
  ...
  connectors: 1
  - name: my-broker
    host: broker
    port: 5672
    routeContainer: true
  linkRoutes: 2
  - prefix: q1
    direction: in
    connection: my-broker
  - prefix: q1
    direction: out
    connection: my-broker
```

1

将此路由器连接到消息代理的连接。Operator 会将这个连接配置从此路由器部署中定义的每个路由器到代理。如果您只在路由器网络和代理间需要一个连接，那么请配置 监听程序而不是连接器，并使代理建立连接。

2

链路路由配置。它定义了传入和传出链接，以及用于将消息应用程序连接到消息代理的连接。

3. 验证路由器是否已建立到消息代理的链接路由。

```
$ oc exec router-mesh-fb6bc5797-crvb6 -it -- qdstat --linkroutes
Link Routes
  address  dir  distrib      status
  ====================================
  q1       in   linkBalanced  active
  q1       out  linkBalanced  active
```

其他资源

- 有关链接路由的更多信息，请参阅创建链接路由。

第 8 章 为消息路由配置地址空间

AMQ Interconnect 提供灵活的应用程序层寻址和交付语义。通过配置地址，您可以在任何广播（接近或均衡）或多播模式中路由消息。

8.1. 在客户端间路由消息

默认情况下，AMQ Interconnect 以均衡的广播模式分发消息（每个消息都传送到单一消费者，AMQ 互联尝试通过网络平衡流量负载）。这意味着，只需要将非默认语义应用到地址或地址范围时才需要更改地址配置。

流程

此流程将地址配置为使用多播分发。路由器网络会将发送到此地址的每个消息的副本分发到订阅了该地址的每个用户。

1. 开始编辑路由器部署的 Interconnect 自定义资源 YAML 文件。

```
$ oc edit -f router-mesh.yaml
```

2. 在 spec 部分中，定义要应用到地址的语义。

router-mesh.yaml 文件示例

```
apiVersion: interconnectedcloud.github.io/v1alpha1
kind: Interconnect
metadata:
  name: router-mesh
spec:
  ...
  addresses:
  - pattern: */orders  (1)
    distribution: multicast
```

(1) 发送到以"订购"结尾的任何地址的消息将以多播模式分发。

Operator 对路由器网络应用更改并重启每个 Pod。

3. 如果您有额外的路由器部署自定义资源，可在路由器网络中定义路由器，请对每个 CR 重复此步骤。

路由器网络中的每个路由器必须具有相同的地址配置。

其他资源

- 有关您可以配置的地址语义的更多信息，请参阅配置消息路由。

8.2. 通过代理路由信息

如果您需要存储和转发消息，您可以在消息代理上通过队列进行路由。在这种情况下，消息制作者向路由器发送消息，路由器将消息发送到代理队列。当消费者连接到路由器来接收消息时，路由器从代理队列检索它们。

您可以将消息路由到与路由器网络相同的 OpenShift 集群中运行的代理，或路由到集群外部运行的代理。

先决条件

- 您必须有权访问消息代理。

流程

1. 开始编辑路由器部署的 Interconnect 自定义资源 YAML 文件。

```
$ oc edit -f router-mesh.yaml
```

2. 在 spec 部分，添加连接器以连接到代理、指向代理的定位点地址，以及 autolinks 来创建到队列的链接。

router-mesh.yaml 文件示例

```
apiVersion: interconnectedcloud.github.io/v1alpha1
kind: Interconnect
metadata:
  name: router-mesh
spec:
  ...
  addresses:
  - prefix: my-queue  1
    waypoint: true
  autoLinks:  2
  - address: my-queue
    direction: in
    connection: my-broker
  - address: my-queue
    direction: out
    connection: my-broker
  connectors:  3
  - name: my-broker
    host: broker
    port: 5672
    routeContainer: true
```

1

哪些消息应存储在代理队列中的地址（或一组地址）。

2

autolink 配置。它定义要用于发送和接收代理的消息的传入和传出链接和连接。

3

将路由器连接到消息代理的连接。

Operator 对路由器网络应用更改并重启每个 Pod。

3. 验证路由器是否已建立到消息代理的 autolink。

```
$ oc exec router-mesh-6d6dccb57f-x5cqf -it -- qdstat --autolinks
AutoLinks
  addr      dir  phs  extAddr  link  status  lastErr
  ================================================
  my-queue  in   1           26    active
  my-queue  out  0           27    active
```

4. 如果您有额外的路由器部署自定义资源，可在路由器网络中定义路由器，请对每个 CR 重复此步骤。

路由器网络中的每个路由器必须具有相同的地址配置。

其他资源

- 有关将消息路由到代理队列和来自代理队列的更多信息，请参阅 通过代理队列路由消息。

第 9 章 使用 PROMETHEUS 和 GRAFANA 监控路由器网络

Prometheus 是为存储历史数据而构建的容器原生虚拟化，用于监控大型可扩展系统，如 AMQ Interconnect。它在延长时间内收集数据，而不仅仅是针对当前运行的会话。

您可以使用 Prometheus 和 Alertmanager 来监控和存储 AMQ Interconnect 数据，以便 Grafana 等图形工具来视觉化并运行对数据的查询。

9.1. 设置 PROMETHEUS 和 GRAFANA

在查看 AMQ Interconnect 仪表板之前，您必须在部署了 AMQ Interconnect 的 OpenShift 项目中部署和配置 Prometheus、Alertmanager 和 Grafana。所有所需的配置文件都在 GitHub 存储库中提供。

流程

1. 克隆qdr -monitoring GitHub 存储库。

   此仓库包含设置 Prometheus 和 Grafana 以监控 AMQ 互联所需的示例配置文件。

   ```
   $ git clone https://github.com/interconnectedcloud/qdr-monitoring
   ```

2. 将 NAMESPACE 环境变量设置为部署 AMQ Interconnect 的项目的名称。

   例如，如果您在示例项目中部署了 AMQ Interconnect，请按如下所示设置 NAMESPACE 环境变量：

   ```
   $ export NAMESPACE=example
   ```

3. 运行 deploy-monitoring.sh 脚本。

   此脚本会创建并配置在 OpenShift 项目中部署 Prometheus、Alertmanager 和 Grafana 所需的 OpenShift 资源。它还配置了两个仪表板，它们为路由器网络提供指标。

   ```
   $ ./deploy-monitoring.sh
   ```

   运行该脚本的另一种方法是将 target 项目指定为参数。例如：

```
$ ./deploy-monitoring.sh example
```

其他资源

- **有关 Prometheus 的更多信息，请参阅 Prometheus 文档。**
- **如需有关 Grafana 的更多信息，请参阅 Grafana 文档。**

9.2. 在 GRAFANA 中查看 AMQ INTERCONNECT 仪表板

设置 Prometheus 和 Grafana 后，您可以在以下 Grafana 仪表板上视觉化 AMQ Interconnect 数据：

Qpid Dispatch Router

显示以下指标：

Qpid Dispatch Router

显示以下指标：

- **deliveries ingress**
- **提供出口**
- **deliveries ingress 路由容器**
- **提供出口路由容器**
- **重定向到回退目的地的交付**
- **丢弃预设定的交付情况**

- 预设定提供的
- auto 链接
- 链接路由
- 地址数
- 连接数
- 链接数

Qpid Dispatch Router - Delayed deliveries

显示以下指标：

- 累计延迟 10 秒
- 累计延迟 1 秒
- 新的延迟交付率

有关这些指标的更多信息，请参阅 第 9.3 节 “路由器指标”。

流程

1. 在 OpenShift web 控制台中，切换到 Networking → Routes，再点击 grafana Route 的 URL。

   Grafana Log In 页面会显示。

2. 输入您的用户名和密码，然后点 Log In。

   默认的 Grafana 用户名和密码都是 admin。第一次登录后，您可以更改密码。

3. 在顶部标题上，单击控制面板下拉菜单，然后选择 Qpid Dispatch Router 或 Qpid Dispatch Router - Delayed Deliveries 仪表板。

图 9.1. 延迟交付仪表板

9.3. 路由器指标

Prometheus 中提供以下指标：

qdr_connections_total

到路由器的网络连接总数。这包括来自 和任何 AMQP 路由容器的连接。

qdr_links_total

附加到路由器的传入和传出链路总数。

qdr_addresses_total

路由器已知的地址总数。

qdr_routers_total

路由器已知的路由器总数。

qdr_link_routes_total

为路由器配置的活跃和不活跃链路路由总数。如需了解更多详细信息，请参阅了解链接路由。

qdr_auto_links_total

为路由器配置的传入和传出自动链路总数。如需有关 auto links 的详情，请参阅配置代理消息传递。

qdr_presettled_deliveries_total

在路由器上交付的预先设置的总数量。路由器设置传入的发送并将设置传播到消息目的地，也称为 *fire 和 forget*。

qdr_dropped_presettled_deliveries_total

预设的交付总数是路由器因拥塞而丢弃的。路由器设置传入的发送并将集合传播到消息目的地，也称为 fire 和 forget。

qdr_accepted_deliveries_total

路由器接受的交付总数。如需有关接受发送的更多信息，请参阅了解消息路由。

qdr_released_deliveries_total

在路由器上发布的交付总数。有关发布的交付的更多信息，请参阅了解消息路由。

qdr_rejected_deliveries_total

路由器中拒绝交付的总数。如需有关拒绝发送的更多信息，请参阅了解消息路由。

qdr_modified_deliveries_total

路由器中修改交付的总数。如需有关修改交付的更多信息，请参阅了解消息路由。

qdr_deliveries_ingress_total

从客户端传送到路由器的消息总数。这包括管理消息，但不包括路由控制消息。

qdr_deliveries_egress_total

从路由器发送到客户端的消息总数。这包括管理消息，但不包括路由控制消息。

qdr_deliveries_transit_total, qdr_deliveries_ingress_route_container_total

通过路由器传递的消息总数，以传送到不同的路由器。

qdr_deliveries_egress_route_container_total

发送到路由器的 AMQP 路由容器的交付总数包括到 AMQ Broker 实例和管理消息的消息，但不路由控制消息。

qdr_deliveries_delayed_1sec_total

路由器转发的交付总数超过一秒钟。

qdr_deliveries_delayed_10sec_total

路由器转发的交付总数超过十秒。

qdr_deliveries_stuck_total

无法交付的总数。通常，交付无法因为 消息路由流控制中缺少贡献的问题

qdr_links_blocked_total

被阻断的链接总数。

qdr_deliveries_redirected_to_fallback_total

转发到回退目的地的交付总数。如需更多信息，请参阅处理 无法传输的信息。

附加信息

请参阅 第 9.2 节 “在 Grafana 中查看 AMQ Interconnect 仪表板”。

第 10 章 使用 AMQ INTERCONNECT WEB 控制台监控路由器网络

您可以使用 AMQ Interconnect Web 控制台来监控路由器网络的状态和性能。默认情况下，当您创建路由器部署时，AMQ Interconnect Operator 会生成用于访问控制台的凭证，并将其存储在 Secret 中。

流程

1. 在 OpenShift 中，切换到 Networking → Routes，再点击控制台 Route。

   Web 控制台在新标签页中打开。

2. 要连接到 Web 控制台，请完成以下字段：

   端口

   输入 443。

   用户名

   输入用户名。

   要查找用于访问 Web 控制台的用户名和密码，请导航至 Workloads → Secrets。包含 web 控制台凭证的 Secret 名为 < *application-name*>-users （如 router-mesh-users）。

   用户名的语法为 <*user*>@< domain >（域是 OpenShift 应用程序名称，这是描述路由器部署的自定义资源的名称）。例如： guest@router-mesh。

   密码

   输入 < *application-name*>-users Secret 中定义的密码。

3. 点 连接。

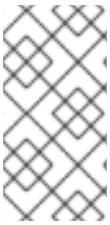
   Router 页面显示路由器网络中的所有路由器。

4. 使用 Web 控制台选项卡监控路由器网络。

此标签页...	provides...
概述	有关路由器、地址、链接、连接和日志的聚合信息。
实体	有关路由器网络中每个路由器的每个 AMQP 管理实体的详细信息。有些属性有您可以添加到 **Charts** 选项卡的图表。
Topology	路由器网络的图形视图，包括路由器、客户端和代理。拓扑显示路由器如何连接，以及如何通过网络传输消息。
charts	在 **实体** 选项卡中选择的信息图表。
消息流	上级图表显示按地址显示实时消息流。
模式	控制路由器网络中各个路由器的管理架构。

第 11 章 MIGRATING FROM RED HAT INTEGRATION - AMQ CERTIFICATE MANAGER OPERATOR

Red Hat Integration - AMQ Certificate Manager Operator 不需要保护您的连接，且 OpenShift Container Platform 4.9 不支持。要在 OpenShift Container Platform 4.9 上安全连接，您可以删除 Red Hat Integration - AMQ Certificate Manager Operator，如下所述。



注意

删除 Red Hat Integration - AMQ Certificate Manager Operator 后，之前发布的证书仍有效，您的连接仍然安全。

先决条件

- **您可以使用 cluster-admin 帐户登录到集群。**

流程

1. **确保已将 Red Hat Integration - AMQ Interconnect Operator 升级到最新版本。**
2. **删除 Red Hat Integration - AMQ Certificate Manager Operator:**
   a. **在 OpenShift Container Platform Web 控制台中导航至 Operators → OperatorHub。**
   b. **从安装的 Operator 列表中选择 Red Hat Integration - AMQ Certificate Manager Operator，然后点 Remove。**
   c. **当显示以下信息时，点 OK ：**

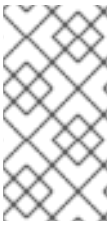
```
Operator Red Hat Integration - AMQ Certificate Manager will be removed from all-
namespaces, but any of its custom resource definitions
or managed resources will remain. If your Operator deployed applications on the cluster
or configured off-cluster resources, these will
continue to run and require manual cleanup.
```

3. **删除 Red Hat Integration - AMQ Certificate Manager Operator CRD**

```
$ oc delete crd issuers.certmanager.k8s.io

customresourcedefinition.apiextensions.k8s.io "issuers.certmanager.k8s.io" deleted
```

4. 验证所有当前连接是否都正常工作。



注意

如果在删除 Red Hat Integration - AMQ Certificate Manager Operator 后创建新连接，您必须按照 第 2.1 节 “为 SSL/TLS 验证创建 secret” 中介绍的步骤来保护新连接。

修订于 2023-01-07 11:27:39 +1000