9.2

IBM MQ dans des conteneurs

Remarque

Avant d'utiliser le présent document et le produit associé, prenez connaissance des informations générales figurant à la section «Remarques», à la page 165.

Cette édition s'applique à la version 9 édition 2 d' IBM® MQ et à toutes les éditions et modifications ultérieures, sauf indication contraire dans les nouvelles éditions.

Table des matières

Multi IBM MQ en conteneurs et IBM Cloud Pak for Integration

Les conteneurs permettent de conditionner un gestionnaire de files d'attente IBM MQ ou une application client IBM MQ avec toutes ses dépendances dans une unité normalisée pour le développement de logiciels.

Vous pouvez exécuter IBM MQ en utilisant le IBM MQ Operator sur Red Hat® OpenShift®. Pour ce faire, utilisez IBM Cloud Pak for Integration, IBM MQ Advanced ou IBM MQ Advanced for Developers.

Vous pouvez aussi exécuter IBM MQ dans un conteneur que vous générez.

MQ Adv. CD Pour plus d'informations sur le IBM MQ Operator, voir les liens suivants :

Multi Planification d'IBM MQ dans des conteneurs

Lors de la planification d'IBM MQ dans des conteneurs, prenez en compte le support fourni par IBM MQ pour diverses options d'architecture, par exemple la façon dont la haute disponibilité est gérée et la manière de sécuriser vos gestionnaires de files d'attente.

Pourquoi et quand exécuter cette tâche

Avant de planifier votre IBM MQ dans l'architecture des conteneurs, vous devez vous familiariser avec les concepts de base de IBM MQ (voir la Présentation technique IBM MQ) ainsi que les concepts Kubernetes/Red Hat OpenShift de base (voir Architecture Red Hat OpenShift Container Platform).

Procédure

- «Comment utiliser IBM MQ dans des conteneurs», à la page 5.
- «Prise en charge de IBM MQ Operator», à la page 6.
- «Prise en charge de la génération de vos propres images de conteneur de gestionnaire de files d'attente IBM MQ», à la page 13.
- «Remarques sur le stockage pour le IBM MQ Operator», à la page 11.
- «Haute disponibilité pour IBM MQ dans les conteneurs», à la page 16.
- «Reprise après incident d'IBM MQ dans des conteneurs», à la page 19.
- «Authentification et autorisation des utilisateurs pour IBM MQ dans les conteneurs», à la page 19.

Comment utiliser IBM MQ dans des conteneurs

Il existe plusieurs options d'utilisation de IBM MQ dans des conteneurs : vous pouvez choisir d'utiliser le IBM MQ Operator, qui utilise des images de conteneur pré-conditionnées, ou créer vos propres images et code de déploiement.

Utilisation de IBM MQ Operator

OpenShift CP4I

Si vous prévoyez un déploiement sur Red Hat OpenShift Container Platform, vous souhaitez probablement utiliser le IBM MQ Operator.

IBM MQ Operator ajoute une nouvelle ressource personnalisée `QueueManager` à Red Hat OpenShift Container Platform. L'opérateur surveille les nouvelles définitions de gestionnaire de files d'attente, puis les transforme en ressources de niveau inférieur nécessaires, telles que les ressources `StatefulSet` et `Service`. Dans le cas de Native HA, l'opérateur peut également effectuer la mise à jour évolutive complexe des instances de gestionnaire de files d'attente. Consultez«Remarques sur l'exécution de votre

propre mise à jour en continu d'un gestionnaire de files d'attente natif de haute disponibilité», à la page 161

Certaines fonctions de IBM MQ ne sont pas prises en charge lors de l'utilisation de IBM MQ Operator. Vous devez générer vos propres images et vos propres chartes si vous voulez effectuer les opérations suivantes :

- Utiliser les API REST pour l'administration ou la messagerie
- Utiliser l'un des composants MQ suivants :
  - Des agents Managed File Transfer et leurs ressources. Toutefois, vous pouvez utiliser IBM MQ Operator pour fournir un ou plusieurs gestionnaires de files d'attente de coordination, de commande ou d'agent.
  - AMQP
  - IBM MQ Bridge to Salesforce
  - IBM MQ Bridge to blockchain (non pris en charge dans les conteneurs)
  - IBM MQ Telemetry Transport (MQTT).
- Personnalisez les options utilisées avec **crtmqm**, **strmqm** et **endmqm**, telles que la configuration des pages de fichier journal. La plupart des options peuvent être configurés à l'aide d'un fichier INI.

Notez que le IBM MQ Operator et les conteneurs évoluent rapidement et ne sont donc pas pris en charge dans certaines éditions de Long Term Support.

Le IBM MQ Operator inclut à la fois des images de conteneur prédéfinies ainsi qu'un code de déploiement pour une exécution sur Red Hat OpenShift Container Platform. Le IBM MQ Operator peut être utilisé pour déployer l'image de conteneur IBM MQ fournie ou une image de conteneur superposée, mais ne peut pas être utilisé pour déployer des images de conteneur MQ intégrées personnalisées.

Génération de vos propres images et code de déploiement

Il s'agit de la solution de conteneur la plus souple, qui exige toutefois de solides compétences relatives à la configuration des conteneurs et qui requiert que vous "possédiez" le conteneur résultant. Si vous ne prévoyez pas d'utiliser Red Hat OpenShift Container Platform, vous devez générer vos propres images et votre propre code de déploiement.

Des exemples de génération d'images sont disponibles. Voir «Génération de votre propre conteneur IBM MQ et code de déploiement», à la page 152.

Concepts associés

«Prise en charge de IBM MQ Operator», à la page 6
Le IBM MQ Operator n'est pris en charge que lorsqu'il est déployé sur Red Hat OpenShift Container Platform.

«Prise en charge de la génération de vos propres images de conteneur de gestionnaire de files d'attente IBM MQ», à la page 13
IBM MQ fournit le code permettant de générer un conteneur de gestionnaire de files d'attente IBM MQ sur GitHub. Cela est basé sur le processus utilisé par IBM pour générer son propre conteneur pris en charge, et vous pouvez utiliser ce référentiel GitHub pour simplifier et accélérer la génération de vos propres images de conteneur.

Prise en charge de IBM MQ Operator

Le IBM MQ Operator n'est pris en charge que lorsqu'il est déployé sur Red Hat OpenShift Container Platform.

IBM MQ Operator utilise des images basées sur des éditions de IBM MQ Continuous Delivery (CD), bien qu'une édition EUS (Extended Update Support) soit disponible avec IBM Cloud Pak for Integration. Les éditions CD sont prises en charge pendant un an ou deux éditions CD (la durée la plus longue est appliquée). Les éditions Long Term Support de IBM MQ ne sont pas disponibles via IBM MQ Operator.

IBM Cloud Pak for Integration 2020.4.1 est une édition EUS (Extended Update Support), qui est prise en charge pendant 18 mois, si vous utilisez une version de IBM MQ marquée comme `-eus`. Sinon, IBM MQ 9.2 est considéré comme une édition Continuous Delivery avec le IBM MQ Operator.

Le IBM MQ Operator utilise des images de conteneur qui fournissent une installation de IBM MQ sur un Red Hat Universal Base Image (UBI), qui inclut des bibliothèques et des utilitaires Linux® clés utilisés par IBM MQ. UBI est pris en charge par Red Hat lorsqu'il est exécuté sous Red Hat OpenShift.

IBM MQ Operator est pris en charge sur les architectures `amd64`et `s390x` (z/Linux).

Concepts associés

«Prise en charge de la génération de vos propres images de conteneur de gestionnaire de files d'attente IBM MQ», à la page 13
IBM MQ fournit le code permettant de générer un conteneur de gestionnaire de files d'attente IBM MQ sur GitHub. Cela est basé sur le processus utilisé par IBM pour générer son propre conteneur pris en charge, et vous pouvez utiliser ce référentiel GitHub pour simplifier et accélérer la génération de vos propres images de conteneur.

OpenShift CP4I CD EUS

Versions prises en charge pour IBM MQ Operator

Mappage entre les versions prises en charge d'IBM MQ, Red Hat OpenShift Container Platform et IBM Cloud Pak for Integration.

Versions IBM MQ disponibles

Canal opérateur	Version de l'opérateur	Versions IBM MQ							
		9.1.5	9.2.0 CD	9.2.0 EUS	9.2.1	9.2.2	9.2.3	9.2.4	9.2.5
v1.6	1.6	⚠	⚠	→	⚠	●	●		
v1.7	1.7	⚠	⚠	→	⚠	●	●	●	
v1.8	1.8	⚠	⚠	→	⚠	⚠	●	●	●

Clé :

● Support Continuous Delivery disponible

▯ Extended Update Support disponible

→ Uniquement disponible lors de la migration à partir du facteur Extended Update Support vers un facteur Continuous Delivery.

⚠ Obsolète. Comme les éditions de IBM MQ ne sont pas pris en charge, elles peuvent encore être configurées dans l'opérateur, mais ne sont plus éligibles pour le support et peuvent être supprimées dans les versions ultérieures.

Pour plus de détails sur chaque version, notamment les fonctionnalités détaillées, les modifications et les correctifs de chaque version, reportez-vous à la rubrique «Historique des éditions de IBM MQ Operator», à la page 21.

Versions Red Hat OpenShift Container Platform compatibles

Canal opérateur	Version de l'opérateur	Versions Red Hat OpenShift Container Platform[1]				
		4.6	4.7[2]	4.8	4.9	4.10
v1.6	1.6	●	●	●	●	●
v1.7	1.7	●	●	●	●	●
v1.8	1.8	●	●	●	●	●

Clé :

● Support Continuous Delivery disponible

▯ Extended Update Support disponible

Versions IBM Cloud Pak for Integration

IBM MQ Operator 1.8.x est pris en charge pour une utilisation en tant que partie de IBM Cloud Pak for Integration version 2021.4.1, ou indépendamment.

IBM MQ Operator 1.7.x est pris en charge pour une utilisation dans le cadre de IBM Cloud Pak for Integration version 2021.4.1, ou indépendamment.

IBM MQ Operator 1.6.x est pris en charge pour une utilisation dans IBM Cloud Pak for Integration version 2021.2.1, 2021.3.1 ou indépendante.

IBM MQ Operator 1.5.x n'est plus pris en charge.

IBM MQ Operator 1.4.x n'est plus pris en charge.

IBM MQ Operator 1.3.x n'est plus pris en charge.

IBM MQ Operator 1.2.x n'est plus pris en charge.

Les IBM MQ Operators 1.1.x et 1.0.x ne sont plus pris en charge.

Versions IBM MQ disponibles dans les anciens opérateurs

Le tableau suivant s'applique aux versions de IBM MQ Operator qui ont désormais atteint leur "fin de vie".

Canal opérateur	Version de l'opérateur	Versions IBM MQ							
		9.1.5	9.2.0 CD	9.2.0 EUS	9.2.1	9.2.2	9.2.3	9.2.4	9.2.5
v1.0	1.0	⚠							
v1.1	1.1	⚠	⚠						
v1.2	1.2	⚠	⚠						

[1] Les versions Red Hat OpenShift Container Platform sont soumises à leurs propres dates pour la prise en charge. Pour plus d'informations, voir Red Hat OpenShift Container Platform Règle de cycle de vie.

[2] IBM MQ Operator dépend de IBM Cloud Pak foundational services. Si vous souhaitez utiliser Red Hat OpenShift Container Platform 4.7, vous devez d'abord mettre à niveau la version de IBM Cloud Pak foundational services.

Canal opérateur	Version de l'opérateur	Versions IBM MQ							
		9.1.5	9.2.0 CD	9.2.0 EUS	9.2.1	9.2.2	9.2.3	9.2.4	9.2.5
v1.3-eus	1.3	⚠	⚠	⚠					
v1.4	1.4	⚠	⚠	→	⚠				
v1.5	1.5	⚠	⚠	→	⚠	⚠			

Clé :

→
Uniquement disponible lors de la migration à partir du facteur Extended Update Support vers un facteur Continuous Delivery.

⚠
Obsolète. Lorsque les éditions de IBM MQ ne sont plus pris en charge, elles peuvent toujours être configurables dans le fichier IBM MQ Operator, mais ne sont plus éligibles pour le support.

Pour plus de détails sur chaque version, notamment les fonctionnalités détaillées, les modifications et les correctifs de chaque version, reportez-vous à la rubrique «Historique des éditions de IBM MQ Operator», à la page 21.

Versions de Red Hat OpenShift Container Platform compatibles avec les opérateurs plus anciens

Le tableau suivant s'applique aux versions de IBM MQ Operator qui ont désormais atteint leur "fin de vie".

Canal opérateur	Version de l'opérateur	Versions Red Hat OpenShift Container Platform[3]						
		4.4[4]	4.5[5]	4.6	4.7[6]	4.8	4.9	4.10
v1.0	1.0	⚠	⚠	⚠	⚠			
v1.1	1.1	⚠	⚠	⚠	⚠	⚠		
v1.2	1.2	⚠	⚠	⚠	⚠	⚠		
v1.3-eus	1.3			⚠	→	→	→	→
v1.4	1.4			⚠	⚠	⚠	⚠	
v1.5	1.5			⚠	⚠	⚠	⚠	⚠

Clé :

[3] Les versions Red Hat OpenShift Container Platform sont soumises à leurs propres dates pour la prise en charge. Pour plus d'informations, voir Red Hat OpenShift Container Platform Règle de cycle de vie.

[4] Red Hat OpenShift Container Platform 4.4 a atteint "la fin de la vie". Pour plus d'informations, voir Red Hat OpenShift Container Platform Règle de cycle de vie.

[5] Red Hat OpenShift Container Platform 4.5 a atteint "la fin de la vie". Pour plus d'informations, voir Red Hat OpenShift Container Platform Règle de cycle de vie.

[6] IBM MQ Operator dépend de IBM Cloud Pak foundational services. Si vous souhaitez utiliser Red Hat OpenShift Container Platform 4.7, vous devez d'abord mettre à niveau la version de IBM Cloud Pak foundational services.

→
Uniquement disponible lors de la migration à partir du facteur Extended Update Support vers un facteur Continuous Delivery.

⚠
La version IBM MQ Operator a atteint sa "fin de vie", mais était déjà disponible sur cette version de Red Hat OpenShift Container Platform

OpenShift CP4I Dépendances pour IBM MQ Operator

IBM MQ Operator est dépendant d'IBM Cloud Pak foundational services Operator, qui installe également l'opérateur IBM Operand Deployment Lifecycle Manager (ODLM). Ces opérateurs seront installés automatiquement lors de l'installation d'IBM MQ Operator. Ces opérateurs dépendants ont une faible empreinte de mémoire et d'unité centrale et sont utilisés pour déployer des ressources supplémentaires dans certaines circonstances.

Lorsque vous créez un `QueueManager`, IBM MQ Operator crée un `OperandRequest` pour les services supplémentaires dont il a besoin. La commande `OperandRequest` est remplie par l'opérateur ODLM, et il installe et instancie les services requis, si nécessaire. Les services requis sont déterminés en fonction du contrat de licence accepté lors du déploiement du gestionnaire de files d'attente et sur lequel les composants du gestionnaire de files d'attente sont demandés.

- Si vous choisissez une licence IBM MQ Advanced ou IBM MQ Advanced for Developers, aucun service supplémentaire n'est demandé. Par exemple, dans le cas suivant, le IBM Cloud Pak foundational services n'est pas utilisé :

```
spec:
  license:
    accept: true
    license: L-APIG-BZDDDY
    use: "Production"
```

- Si vous choisissez une licence IBM Cloud Pak for Integration et que vous choisissez d'activer le serveur Web, IBM MQ Operator instanciera également l'opérateur IBM Identity and Access Management (IAM) pour activer la connexion unique. IAM Operator est déjà disponible si vous avez installé IBM Cloud Pak for Integration Operator. Exemple :

```
spec:
  license:
    accept: true
    license: L-RJON-BUVMQX
    use: "Production"
```

  Toutefois, si vous désactivez le serveur Web, aucun IBM Cloud Pak foundational services n'est demandé. Exemple :

```
spec:
  license:
    accept: true
    license: L-RJON-BUVMQX
    use: "Production"
  web:
    enabled: false
```

Les anciennes versions de IBM MQ Operator ont toujours demandé l'installation de IBM Licensing Operator (et de ses dépendances) afin de suivre l'utilisation des licences. À partir de IBM MQ Operator 1.5, le service de licence n'est pas demandé et vous devez le demander séparément.

IBM MQ Operator requiert 1 coeur d'UC et 1 Go de mémoire. Pour obtenir une ventilation détaillée des configurations matérielle et logicielle requises pour les opérateurs dépendants, voir Exigences matérielles et recommandations pour les services de base.

Vous pouvez choisir la quantité d'unité centrale et de mémoire utilisée par vos gestionnaires de file d'attente. Pour plus d'informations, voir «.spec.queueManager.resources», à la page 141.

Référence associée

«Référence relative à l'octroi de licence pour mq.ibm.com/v1beta1», à la page 131

OpenShift CP4I Droits d'accès au cluster requis par IBM MQ Operator

IBM MQ Operator requiert des droits d'accès au cluster pour gérer les webhooks d'admission et les exemples, ainsi que pour lire les informations de la classe de stockage et de la version du cluster.

IBM MQ Operator requiert les droits d'accès au cluster suivants :

- Droit de gérer les webhooks d'admission. Permet de créer, d'extraire et de mettre à jour des crochets spécifiques utilisés dans le processus de création et de gestion des conteneurs fournis par l'opérateur.
  - Groupes d'API : **admissionregistration.k8s.io**
  - Ressources : **validatingwebhookconfigurations**
  - Instructions : **create, get, update**
- Permet de créer et de gérer des ressources utilisées dans la console Red Hat OpenShift pour fournir des exemples et des fragments lors de la création de ressources personnalisées.
  - Groupes d'API : **console.openshift.io**
  - Ressources : **consoleyamlsamples**
  - Instructions : **create, get, update, delete**
- Droit de lecture de la version du cluster. Permet à l'opérateur de faire part de tout problème concernant l'environnement du cluster.
  - Groupes d'API : **config.openshift.io**
  - Ressources : **clusterversions**
  - Instructions : **get, list, watch**
- Droit de lecture des classes de stockage sur le cluster. Permet à l'opérateur de faire part de tout problème concernant certaines classes de stockage dans les conteneurs.
  - Groupes d'API : **storage.k8s.io**
  - Ressources : **storageclasses**
  - Instructions : **get, list**

OpenShift CP4I Kubernetes Remarques sur le stockage pour le IBM MQ Operator

Le IBM MQ Operator peut être exécuté dans deux modes de stockage :

- L'option **Stockage éphémère** est utilisée lorsque toutes les informations d'état du conteneur peuvent être supprimées lors du redémarrage du conteneur. En général, il est utilisé lorsque des environnements sont créés à des fins de démonstration ou lors d'un développement avec des gestionnaires de files d'attente autonomes.
- Le **stockage persistant** constitue la configuration courante pour IBM MQ et garantit que si le conteneur est redémarré, la configuration, les journaux et les messages persistants existants seront disponibles dans le conteneur redémarré.

IBM MQ Operator permet de personnaliser les caractéristiques de stockage qui peuvent différer considérablement selon l'environnement, ainsi que le mode de stockage souhaité.

Stockage éphémère

IBM MQ est une application avec état et elle conserve cet état dans le stockage en vue d'une reprise en cas de redémarrage. Si vous utilisez le stockage éphémère, toutes les informations d'état du gestionnaire de files d'attente sont perdues au redémarrage. Seront perdus :

- Tous les messages
- Toutes les informations relatives à l'état des communications entre les gestionnaires de files d'attente (numéros de séquence des messages de canal)
- Identité du cluster MQ du gestionnaire de files d'attente
- Toutes les informations relatives à l'état des transactions
- L'intégralité de la configuration du gestionnaire de files d'attente
- Toutes les données de diagnostic locales

Ainsi, vous devez déterminer si le stockage éphémère est approprié pour un scénario de production, de test ou de développement, par exemple lorsque tous les messages sont non persistants et que le gestionnaire de files d'attente n'est pas membre d'un cluster MQ. En plus de disposer de tous les états de messagerie au redémarrage, la configuration du gestionnaire de files d'attente est également supprimée. Pour obtenir un conteneur intégralement éphémère, vous devez ajouter la configuration d'IBM MQ à l'image de conteneur (pour plus d'informations, voir «Génération d'une image avec des fichiers MQSC et INI personnalisés, à l'aide de l'interface de ligne de commande Red Hat OpenShift», à la page 117). Sinon, IBM MQ devra être configuré à chaque fois que le conteneur redémarre.

Par exemple, pour configurer IBM MQ avec un stockage éphémère, le type de stockage de `QueueManager` doit inclure les éléments suivants :

```
queueManager:
  storage:
    queueManager:
      type: ephemeral
```

Stockage de persistance

Normalement, IBM MQ s'exécute avec un stockage persistant afin de garantir que les messages persistants et la configuration du gestionnaire de files d'attente sont conservés après un redémarrage. Il s'agit donc du comportement par défaut. Etant donné les divers fournisseurs de stockage et les différentes fonctions que chaque fournisseur prend en charge, cela signifie souvent qu'il est nécessaire de personnaliser la configuration. L'exemple ci-après présente les zones communes permettant de personnaliser la configuration de stockage de MQ dans l'API v1beta1 :

- spec.queueManager.availability contrôle le mode de disponibilité. Si vous utilisez `SingleInstance`, le stockage `ReadWriteOnce` est suffisant, alors que `multiInstance` requiert une classe d'archivage qui prend en charge `ReadWriteMany` avec les caractéristiques de verrouillage de fichier appropriées. IBM MQ fournit une déclaration de prise en charge et une déclaration de test. Le mode de disponibilité a également un impact sur la présentation des volumes persistants. Pour plus d'informations, voir «Haute disponibilité pour IBM MQ dans les conteneurs», à la page 16.
- spec.queueManager.storage contrôle les paramètres de stockage individuels. Un gestionnaire de files d'attente peut être configuré pour utiliser entre un et quatre volumes persistants.

L'exemple suivant est un fragment de configuration simple qui utilise un gestionnaire de files d'attente mono-instance :

```
spec:
  queueManager:
    storage:
      queueManager:
        enabled: true
```

L'exemple suivant est un fragment de configuration de gestionnaire de files d'attente multi-instance, qui présente une classe d'archivage autre que la classe d'archivage par défaut, ainsi qu'un stockage de fichiers nécessitant des groupes supplémentaires :

```
spec:
  queueManager:
    availability:
```

```
      type: MultiInstance
    storage:
      queueManager:
        class: ibmc-file-gold-gid
      persistedData:
        enabled: true
        class: ibmc-file-gold-gid
      recoveryLogs:
        enabled: true
        class: ibmc-file-gold-gid
  securityContext:
     supplementalGroups: [99]
```

Remarque : Vous pouvez également configurer des groupes supplémentaires avec des gestionnaires de files d'attente à instance unique.

Remarque : Vous n'avez pas besoin de systèmes de fichiers partagés si vous utilisez Native HA (voir «Haute disponibilité pour IBM MQ dans les conteneurs», à la page 16). En particulier, vous ne devez pas utiliser NFSv3.

Linux Prise en charge de la génération de vos propres images de conteneur de gestionnaire de files d'attente IBM MQ

IBM MQ fournit le code permettant de générer un conteneur de gestionnaire de files d'attente IBM MQ sur GitHub. Cela est basé sur le processus utilisé par IBM pour générer son propre conteneur pris en charge, et vous pouvez utiliser ce référentiel GitHub pour simplifier et accélérer la génération de vos propres images de conteneur.

Le code est fourni dans le référentiel mq-container GitHub ici : https://github.com/ibm-messaging/mq-container. Il est fourni sous licence Apache 2.0, avec le support fourni par la communauté.

Le référentiel n'utilise pas les modules Linux rpm standard ; il utilise le package compressé pour les déploiements de conteneur. L'avantage de cette approche est que vous pouvez exécuter dans des environnements de conteneur plus sécurisés sans avoir besoin de droits d'accès plus importants. Toutefois, cela a une incidence sur les options de sécurité disponibles, car IBM MQ utilise traditionnellement les droits d'accès plus importants pour l'authentification basée sur le système d'exploitation. Pour un déploiement de conteneur, l'utilisation de l'authentification basée sur le système d'exploitation n'est normalement pas une bonne pratique ; vous pouvez plutôt utiliser l'authentification mutuelle TLS ou LDAP. Avec IBM MQ Advanced for Developers, vous pouvez également utiliser l'authentification par fichier, ce qui permet à vos utilisateurs de démarrer rapidement.

Le gestionnaire de files d'attente de données répliquées (RDQM) n'est pas pris en charge dans un environnement de conteneur. Vous pouvez obtenir des fonctions similaires à RDQM à l'aide de «Native HA», à la page 96.

Concepts associés

«Prise en charge de IBM MQ Operator», à la page 6
Le IBM MQ Operator n'est pris en charge que lorsqu'il est déployé sur Red Hat OpenShift Container Platform.

IBM MQ -Images de non-installation

Linux Annotations de licence lors de la génération de votre propre image de conteneur IBM MQ

Les annotations de licence vous permettent de suivre l'utilisation en fonction des limites définies sur le conteneur, plutôt que sur la machine sous-jacente. Vous configurez vos clients pour déployer le conteneur avec des annotations spécifiques que IBM License Service utilise ensuite pour suivre l'utilisation.

Lors du déploiement d'une image de conteneur IBM MQ auto-générée, il existe deux approches communes pour l'octroi de licences :

- Licence sur l'ensemble de la machine qui exécute le conteneur.
- Licence sur le conteneur en fonction des limites associées.

Les deux options sont disponibles pour les clients, et des détails supplémentaires sont disponibles sur la page des licences de conteneur IBM sous Passport Advantage.

Si le conteneur IBM MQ doit être concédé sous licence en fonction des limites de conteneur, IBM License Service doit être installé pour suivre l'utilisation. Pour plus d'informations sur les environnements pris en charge et les instructions d'installation, consultez la page ibm-licensing-operator sur GitHub.

IBM License Service est installé sur le cluster Kubernetes où le conteneur IBM MQ est déployé, et les annotations de pod sont utilisées pour le suivi de l'utilisation. Par conséquent, les clients doivent déployer le pod avec des annotations spécifiques que IBM License Service utilise ensuite. En fonction de vos droits et de vos capacités déployées dans le conteneur, utilisez une ou plusieurs des annotations suivantes :

Conteneur IBM MQ Advanced

```
productName: "IBM MQ Advanced"
productID: "208423bb063c43288328b1d788745b0c"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "PROCESSOR_VALUE_UNIT" |  "VIRTUAL_PROCESSOR_CORE"
```

Conteneur IBM MQ Advanced High Availability Replica

```
productName: "IBM MQ Advanced High Availability Replica"
productID: "546cb719714942c18748137ddd8d5659"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "PROCESSOR_VALUE_UNIT" |  "VIRTUAL_PROCESSOR_CORE"
```

Conteneur IBM MQ Base

```
productName: "IBM MQ"
productID: "c661609261d5471fb4ff8970a36bccea"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "PROCESSOR_VALUE_UNIT" |  "VIRTUAL_PROCESSOR_CORE"
```

Conteneur IBM MQ Base High Availability Replica

```
productName: "IBM MQ High Availability Replica"
productID: "2a2a8e0511c849969d2f286670ea125e"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "PROCESSOR_VALUE_UNIT" |  "VIRTUAL_PROCESSOR_CORE"
```

Conteneur IBM MQ Advanced for Developers

```
productName: "IBM MQ Advanced for Developers"
productID: "2f886a3eefbe4ccb89b2adb97c78b9cb"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "FREE"
```

Conteneur IBM MQ Advanced avec une autorisation d'utilisation CP4I (production)

```
productName: "IBM MQ Advanced with CP4I License"
productID: "208423bb063c43288328b1d788745b0c"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "VIRTUAL_PROCESSOR_CORE"
productCloudpakRatio: "2:1"
cloudpakName: "IBM Cloud Pak for Integration"
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
```

Conteneur IBM MQ Advanced High Availability Replica avec une autorisation d'utilisation CP4I (production)

```
productName: "IBM MQ Advanced High Availability Replica with CP4I License"
productID: "546cb719714942c18748137ddd8d5659"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "VIRTUAL_PROCESSOR_CORE"
productCloudpakRatio: "10:1"
cloudpakName: "IBM Cloud Pak for Integration"
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
```

Conteneur IBM MQ Advanced avec une autorisation d'utilisation CP4I (hors production)

```
productName: "IBM MQ Advanced for Non-Production with CP4I License"
productID: "21dfe9a0f00f444f888756d835334909"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "VIRTUAL_PROCESSOR_CORE"
productCloudpakRatio: "4:1"
cloudpakName: "IBM Cloud Pak for Integration"
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
```

Conteneur IBM MQ Advanced High Availability Replica avec une autorisation d'utilisation CP4I (hors production)

```
productName: "IBM MQ Advanced High Availability Replica for Non-Production with CP4I License"
productID: "b3f8f984007d47fb981221589cc50081"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "VIRTUAL_PROCESSOR_CORE"
productCloudpakRatio: "20:1"
cloudpakName: "IBM Cloud Pak for Integration"
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
```

Conteneur IBM MQ Base avec une autorisation d'utilisation CP4I (production)

```
productName: "IBM MQ with CP4I License"
productID: "c661609261d5471fb4ff8970a36bccea"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "VIRTUAL_PROCESSOR_CORE"
productCloudpakRatio: "4:1"
cloudpakName: "IBM Cloud Pak for Integration"
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
```

Conteneur IBM MQ Base High Availability Replica avec une autorisation d'utilisation CP4I (production)

```
productName: "IBM MQ High Availability Replica with CP4I License"
productID: "2a2a8e0511c849969d2f286670ea125e"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "VIRTUAL_PROCESSOR_CORE"
productCloudpakRatio: "20:1"
cloudpakName: "IBM Cloud Pak for Integration"
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
```

Conteneur IBM MQ Base avec une autorisation d'utilisation CP4I (hors production)

```
productName: "IBM MQ with CP4I License Non-Production"
productID: "151bec68564a4a47a14e6fa99266deff"
productChargedContainers:  "All" | "NAME_OF_CONTAINER"
productMetric: "VIRTUAL_PROCESSOR_CORE"
productCloudpakRatio: "8:1"
cloudpakName: "IBM Cloud Pak for Integration"
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
```

Conteneur IBM MQ Base High Availability Replica avec une autorisation d'utilisation CP4I (hors production)

```
productName: "IBM MQ High Availability Replica with CP4I License Non-Production"
productID: "f5d0e21c013c4d4b8b9b2ce701f31928"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "VIRTUAL_PROCESSOR_CORE"
productCloudpakRatio: "40:1"
cloudpakName: "IBM Cloud Pak for Integration"
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
```

OpenShift CP4I Kubernetes Haute disponibilité pour IBM MQ dans les conteneurs

Trois choix s'offrent à vous pour la haute disponibilité avec IBM MQ Operator : **le gestionnaire de files d'attente Native HA** (qui dispose d'un serveur secondaire actif et deux serveurs secondaires de secours), **le gestionnaire de files d'attente multi-instance** (qui est une paire active-veille utilisant un système de fichiers partagé en réseau) ou **le gestionnaire de files d'attente résilient unique** (qui offre une approche simple pour la haute disponibilité à l'aide du stockage en réseau). Les deux derniers s'appuient sur le système de fichiers pour assurer la disponibilité des données récupérables, contrairement à Native HA. Par conséquent, lorsque vous n'utilisez pas Native HA, la disponibilité du système de fichiers est essentielle à la disponibilité du gestionnaire de files d'attente. Si la récupération des données est importante, le système de fichiers doit assurer la redondance via la réplication.

Vous devez envisager la disponibilité des **messages** et la disponibilité des **services** séparément. Avec IBM MQ for Multiplatforms, un message est stocké dans un gestionnaire de files d'attente et un seul. Ainsi, si ce gestionnaire de files d'attente n'est plus disponible, vous perdez temporairement l'accès aux messages qu'il contient. Pour que les messages soient hautement disponibles, vous devez être

capable de récupérer un gestionnaire de files d'attente aussi vite que possible. Vous pouvez assurer la disponibilité des `services` en créant plusieurs instances des files d'attente que les applications client pourront utiliser, par exemple à l'aide d'un cluster uniforme IBM MQ.

Vous pouvez considérer qu'un gestionnaire de files d'attente est composé de deux parties : les données stockées sur disque et les processus en cours d'exécution qui permettent d'accéder aux données. Vous pouvez déplacer tout gestionnaire de files d'attente sur un noeud Kubernetes différent, tant que le noeud conserve les mêmes données (fournies par des volumes Kubernetes persistants) et qu'il peut être associé à une adresse sur le réseau par les applications client. Dans Kubernetes, un service est utilisé pour fournir une identité réseau cohérente.

IBM MQ s'appuie sur la disponibilité des données sur les volumes persistants. Par conséquent, la disponibilité du stockage fournissant les volumes persistants est critique pour la disponibilité du gestionnaire de files d'attente, car IBM MQ ne peut pas être plus disponible que le stockage qu'il utilise. Si vous décidez de tolérer l'indisponibilité d'une zone de disponibilité entière, vous devez utiliser un fournisseur de volumes qui réplique les écritures sur disque dans une autre zone.

Gestionnaire de files d'attente Native HA

Les gestionnaires de files d'attente Native HA sont disponibles à partir de IBM Cloud Pak for Integration 2021.2.1, en utilisant IBM MQ Operator 1.6 ou versions supérieures, avec IBM MQ 9.2.3 ou versions supérieures.

Les gestionnaires de files d'attente Native HA impliquent un **actif** et deux pods **réplique** Kubernetes , qui s'exécutent dans le cadre d'un Kubernetes StatefulSet avec exactement trois répliques chacune avec leur propre ensemble de volumes persistants Kubernetes . Les exigences IBM MQ pour les systèmes de fichiers partagés s'appliquent également lors de l'utilisation d'un gestionnaire de files d'attente Native HA (à l'exception du verrouillage basé sur bail), mais vous n'avez pas besoin d'utiliser un système de fichiers partagé. Vous pouvez utiliser le stockage par blocs, avec un système de fichiers adapté, comme *xfs* ou *ext4*. Les temps de reprise d'un gestionnaire de files d'attente Native HA sont contrôlés par les facteurs suivants :

1. Le temps nécessaire aux répliques d'instances pour détecter que l'instance active a échoué. Ce facteur peut être configuré.
2. Le temps nécessaire à la sonde de vigilance du pod Kubernetes pour détecter si le conteneur prêt a changé et rediriger le trafic réseau. Ce facteur peut être configuré.
3. Le temps nécessaire aux clients IBM MQ pour se reconnecter.

Pour plus d'informations, voir «Native HA», à la page 96

Gestionnaire de files d'attente multi-instance

Les gestionnaires de files d'attente multi-instance impliquent un pod **actif** et un pod **en veille** Kubernetes, qui s'exécutent en tant que partie d'un ensemble de statistiques Kubernetes avec exactement deux répliques et un ensemble de volumes persistants Kubernetes. Les données et les journaux des transactions du gestionnaire de files d'attente sont conservés sur deux volumes persistants à l'aide d'un système de fichiers partagé.

Les gestionnaires de files d'attente multi-instances exigent que le pod **actif** et le pod **de secours** disposent d'un accès simultané au volume persistant. Pour configurer cet accès, vous utilisez des volumes Kubernetes persistants pour lesquels le mode d'accès (paramètre **`access mode`**) est `ReadWriteMany`. Les volumes doivent également répondre aux IBM MQ exigences pour les systèmes de fichiers partagés, car IBM MQ s'appuie sur la libération automatique des verrous de fichier pour déclencher une reprise en ligne du gestionnaire de files d'attente. IBM MQ fournit une liste de systèmes de fichiers testés.

Les temps de reprise pour un gestionnaire de files d'attente multi-instance dépendent des facteurs suivants :

1. Le temps nécessaire au système de fichiers partagé, après un échec, pour libérer les verrous initialement placés par l'instance active.
2. Le temps nécessaire à l'instance de secours pour acquérir les verrous, puis démarrer.
3. Le temps nécessaire à la sonde de vigilance du pod Kubernetes pour détecter si le conteneur prêt a changé et rediriger le trafic réseau. Ce facteur peut être configuré.
4. Le temps nécessaire aux clients IBM MQ pour se reconnecter.

Gestionnaire de files d'attente résilient unique

Un gestionnaire de files d'attente résilient unique est une instance unique d'un gestionnaire de files d'attente qui s'exécute dans un pod Kubernetes unique, où Kubernetes surveille le gestionnaire de files d'attente et remplace le pod si nécessaire.

Les IBM MQ exigences pour les systèmes de fichiers partagés s'appliquent également lors de l'utilisation d'un seul gestionnaire de files d'attente résilient (sauf pour le verrouillage basé sur un bail), mais vous n'avez pas besoin d'utiliser un système de fichiers partagé. Vous pouvez utiliser le stockage par blocs, avec un système de fichiers adapté, comme *xfs* ou *ext4*.

Les temps de reprise pour un gestionnaire de files d'attente résilient unique dépendent des facteurs suivants :

1. Le temps d'exécution de la sonde de non-défaillance et le nombre d'échecs qu'elle tolère. Ce facteur peut être configuré.
2. Le temps nécessaire au planificateur Kubernetes pour replanifier le pod défectueux sur un nouveau noeud.
3. Le temps nécessaire pour télécharger l'image de conteneur sur le nouveau noeud. Si vous avez associé le paramètre **`imagePullPolicy`** à la valeur `IfNotPresent`, il se peut que l'image soit déjà disponible sur ce noeud.
4. Le temps nécessaire à la nouvelle instance de gestionnaire de files d'attente pour démarrer.
5. Le temps nécessaire à la sonde de vigilance du pod Kubernetes pour détecter si le conteneur est prêt. Ce facteur peut être configuré.
6. Le temps nécessaire aux clients IBM MQ pour se reconnecter.

Important :

Bien que le modèle de gestionnaire de files d'attente résilient unique présente certains avantages, vous devez déterminer si vous pouvez atteindre vos objectifs de disponibilité avec les limitations liées aux échecs de noeud.

Dans Kubernetes, un pod défectueux est généralement récupéré rapidement, mais l'échec d'un noeud entier est traité différemment. Lorsque vous utilisez une charge de travail avec état telle que IBM MQ avec un objet Kubernetes StatefulSet, si un noeud maître Kubernetes perd le contact avec un noeud worker, il ne peut pas déterminer si le noeud a échoué ou s'il a simplement perdu la connectivité du réseau. Ainsi, Kubernetes n'effectue **aucune action** dans ce cas, sauf si l'un des événements suivants survient :

1. Le noeud est restauré dans un état permettant la communication avec le noeud principal Kubernetes.
2. Une action d'administration est effectuée pour supprimer explicitement le pod sur le noeud Kubernetes principal. Elle n'arrête pas nécessairement l'exécution du pod, mais le supprime du magasin Kubernetes. Par conséquent, l'action d'administration doit être utilisée avec précaution.

Tâches associées

«Configuration de la haute disponibilité pour les gestionnaires de files d'attente à l'aide de IBM MQ Operator», à la page 96

Référence associée

Configurations à haute disponibilité

OpenShift CP4I Kubernetes Reprise après incident d'IBM MQ dans des conteneurs

Vous devez prendre en compte le type de sinistre pour lequel vous vous préparez. Dans les environnements cloud, l'utilisation de zones de disponibilité offre un certain niveau de tolérance aux sinistres et est plus conviviale. Si vous disposez d'un nombre impair de centres de données (pour le quorum) et d'une liaison réseau à faible latence, vous pouvez éventuellement exécuter un cluster Red Hat OpenShift Container Platform ou Kubernetes unique avec plusieurs zones de disponibilité, chacune dans un emplacement physique distinct. Cette rubrique aborde les points à prendre en compte pour la reprise après incident lorsque ces critères ne peuvent pas être respectés, à savoir, si le nombre de centres de données est pair ou que la liaison réseau est à latence élevée.

Pour la reprise après incident, vous devez prendre en compte les points suivants :

- Réplication de données IBM MQ (conservées dans une ou plusieurs ressources `PersistentVolume`) dans l'emplacement de reprise après incident
- Nouvelle création du gestionnaire de files d'attente à l'aide des données répliquées
- ID réseau du gestionnaire de files d'attente visible par les applications clientes IBM MQ et les autres gestionnaires de files d'attente. Cet ID peut être une entrée du serveur de noms de domaine, par exemple.

Les données persistantes doivent être répliquées, de manière synchrone ou asynchrone, sur le site de reprise après incident. Ceci est généralement spécifique au fournisseur de stockage, mais peut également être effectué à l'aide d'un `VolumeSnapshot`. Pour plus d'informations sur les instantanés de volume, reportez-vous à la rubrique CSI volume snapshots.

Lors d'une reprise après incident, vous devez recréer l'instance de gestionnaire de files d'attente sur le nouveau cluster Kubernetes, à l'aide des données répliquées. Si vous utilisez IBM MQ Operator, vous aurez besoin de `QueueManager` YAML, ainsi que de YAML pour d'autres ressources de prise en charge telles que `ConfigMap` ou `Secret`.

Information associée

ha_for_ctr.dita

Authentification et autorisation des utilisateurs pour IBM MQ dans les conteneurs

IBM MQ peut être configuré pour utiliser des utilisateurs et des groupes LDAP. Vous pouvez également utiliser des utilisateurs et des groupes du système d'exploitation local dans l'image de conteneur. Le IBM MQ Operator n'autorise pas l'utilisation des utilisateurs et des groupes du système d'exploitation, en raison de problèmes de sécurité.

Dans un environnement conteneurisé à service partagé, des contraintes de sécurité sont généralement mises en place pour éviter tout problème de sécurité. Par exemple :

- **Empêcher l'utilisation de l'utilisateur "root" dans un conteneur**
- **Forcer l'utilisation d'un ID utilisateur aléatoire**. Par exemple, dans Red Hat OpenShift Container Platform, l'objet `SecurityContextConstraints` par défaut (appelé `restricted`) utilise un ID utilisateur par conteneur.
- **Empêcher l'utilisation de l'escalade des privilèges**. IBM MQ sous Linux utilise l'escalade des privilèges pour vérifier les mots de passe des utilisateurs ; il utilise un programme "setuid" afin de devenir l'utilisateur "root" pour cela.

OpenShift CP4I Pour garantir le respect de ces mesures de sécurité, le IBM MQ Operator n'autorise pas l'utilisation d'ID définis dans les bibliothèques du système d'exploitation dans un conteneur. Aucun ID utilisateur ou groupe `mqm` n'est défini dans le conteneur. Lorsque vous utilisez IBM MQ dans IBM Cloud Pak for Integration et Red Hat OpenShift, vous devez configurer votre gestionnaire de file d'attente pour que celui-ci utilise l'authentification et l'autorisation des utilisateurs. Pour des

informations sur la configuration d'IBM MQ dans cette optique, voir Connection authentication: User repositories et LDAP authorization.

Multi Planification de l'évolutivité et des performances pour IBM MQ dans les conteneurs

Dans la plupart des cas, la mise à l'échelle et les performances d' IBM MQ dans les conteneurs sont identiques à celles d' IBM MQ for Multiplatforms. Toutefois, quelques limites supplémentaires peuvent être imposées par la plateforme de conteneurs.

Pourquoi et quand exécuter cette tâche

Lorsque vous planifiez l'évolutivité et les performances d' IBM MQ dans des conteneurs, tenez compte des options suivantes:

Procédure

- **Limiter le nombre d'unités d'exécution et de processus.**

  IBM MQ utilise des unités d'exécution pour gérer les accès concurrents. Dans Linux, les unités d'exécution sont implémentées en tant que processus, de sorte que vous pouvez rencontrer des limites imposées par la plateforme de conteneur ou le système d'exploitation, sur le nombre maximal de processus. Dans Red Hat OpenShift Container Platform, il existe une limite par défaut de 4096 processus par conteneur (1024 processus jusqu'à OpenShift 4.11). Bien que cela soit approprié pour la grande majorité des scénarios, cela peut avoir un impact sur le nombre de connexions client pour un gestionnaire de files d'attente.

  La limite de processus dans Kubernetes peut être configurée par un administrateur de cluster à l'aide du paramètre de configuration kubelet **`podPidsLimit`**. Voir Process ID limits and reservation dans la documentation Kubernetes . Dans Red Hat OpenShift Container Platform, vous pouvez également créer une ressource personnalisée **`ContainerRuntimeConfig`** pour éditer les paramètres CRI-O.

  Dans votre configuration IBM MQ , vous pouvez également définir le nombre maximal de connexions client pour un gestionnaire de files d'attente. Voir Limites de canal de connexion serveur pour l'application de limites à un canal de connexion serveur individuel et l' attribut MAXCHANNELS INI pour l'application de limites à l'ensemble du gestionnaire de files d'attente.
- **Limiter le nombre de volumes.**

  Dans les systèmes de cloud et de conteneur, les volumes de stockage connectés au réseau sont couramment utilisés. Le nombre de volumes pouvant être connectés à des noeuds Linux est limité. Par exemple, AWS EC2 ne limite pas plus de 30 volumes par machine virtuelle. Red Hat OpenShift Container Platform a une limite similaire, tout comme Microsoft Azure et Google Cloud Platform.

  Un gestionnaire de files d'attente Native HA requiert un volume pour chacune des trois instances et applique les instances à répartir entre les noeuds. Toutefois, vous pouvez configurer le gestionnaire de files d'attente pour qu'il utilise trois volumes par instance (données du gestionnaire de files d'attente, journaux de reprise et données persistantes).
- **Utiliser les techniques de mise à l'échelle IBM MQ .**

  Au lieu d'utiliser un petit nombre de gestionnaires de files d'attente volumineux, il peut être utile d'utiliser des techniques de mise à l'échelle IBM MQ telles que des clusters uniformes IBM MQ pour exécuter plusieurs gestionnaires de files d'attente avec la même configuration. Ceci a pour avantage supplémentaire de réduire l'impact d'un redémarrage d'un conteneur unique (par exemple, dans le cadre de la maintenance de la plateforme de conteneur).

OpenShift CP4I CD EUS Utilisation de IBM MQ dans IBM Cloud Pak for Integration et Red Hat OpenShift

Le IBM MQ Operator déploie et gère IBM MQ dans le cadre de IBM Cloud Pak for Integration, ou de manière autonome sur Red Hat OpenShift Container Platform

Procédure

- «Historique des éditions de IBM MQ Operator», à la page 21.
- «Migration d'IBM MQ vers IBM Cloud Pak for Integration», à la page 37.
- «Installation et désinstallation de IBM MQ Operator sous Red Hat OpenShift», à la page 62.
- «Mise à niveau de IBM MQ Operator et des gestionnaires de files d'attente», à la page 74.
- «Déploiement et configuration de gestionnaires de files d'attente à l'aide de IBM MQ Operator», à la page 83.
- «Utilisation de IBM MQ à l'aide de IBM MQ Operator», à la page 120.
- «Référence d'API pour IBM MQ Operator», à la page 131.

OpenShift CP4I CD EUS Historique des éditions de IBM MQ Operator

IBM MQ Operator

IBM MQ Operator 1.8.2

CD

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2021.4.1

Canal opérateur
`v1.8`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.0-r3, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus, 9.2.0.2-r2-eus, 9.2.0.5-r1-eus, 9.2.0.5-r2-eus, 9.2.0.5-r3-eus, 9.2.1.0-r1, 9.2.1.0-r2, 9.2.2.0-r1, 9.2.3.0-r1, 9.2.4.0-r1, 9.2.5.0-r1, 9.2.5.0-r2, 9.2.5.0-r3

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 et versions ultérieures

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services3.8 et version supérieure (canal v3)

Nouveautés

- Mise à jour de sécurité uniquement basée sur IBM MQ Operator 1.8.0.
- Les vulnérabilités qui sont prises en compte sont détaillées dans ce bulletin de sécurité.

IBM MQ Operator 1.8.1

CD

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2021.4.1

Canal opérateur
`v1.8`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.0-r3, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus, 9.2.0.2-r2-eus, 9.2.1.0-r1, 9.2.1.0-r2, 9.2.2.0-r1, 9.2.3.0-r1, 9.2.4.0-r1, 9.2.5.0-r1, 9.2.5.0-r2

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 et versions ultérieures

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services3.8 et version supérieure (canal v3)

Nouveautés

- Mise à jour de sécurité uniquement basée sur IBM MQ Operator 1.8.0.
- Les vulnérabilités qui sont prises en compte sont détaillées dans ce bulletin de sécurité.

IBM MQ Operator 1.8.0

CD

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2021.4.1

Canal opérateur
`v1.8`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.0-r3, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus, 9.2.0.2-r2-eus, 9.2.1.0-r1, 9.2.1.0-r2, 9.2.2.0-r1, 9.2.3.0-r1, 9.2.4.0-r1, 9.2.5.0-r1

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 et versions ultérieures

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services3.8 et version supérieure (canal v3)

Nouveautés

- Ajoute les conditions de statut des versions obsolètes de IBM MQ.

Modifications

- Les images sont déplacées de Docker Hub vers IBM Container Registry.
  - Les clients disposant de règles de pare-feu peuvent avoir besoin de les ajuster pour accéder aux images sous IBM Container Registry.
  - Les clients Airgap connaissent un redémarrage de noeud lors de la mise à niveau vers IBM MQ Operator 1.8.0.
- Versions obsolètes : IBM MQ 9.1.5, 9.2.0 CD, 9.2.1, 9.2.2. Il se peut que ces versions ne soient pas réconciliées par les versions ultérieures de IBM MQ Operator.
- Modifications de la logique de licence : la mise à niveau des clients vers IBM MQ 9.2.5 ne peut utiliser que les licences spécifiées pour fonctionner avec IBM MQ 9.2.5. Voir «Référence relative à l'octroi de licence pour mq.ibm.com/v1beta1», à la page 131.
- Les vulnérabilités qui sont prises en compte sont détaillées dans ce bulletin de sécurité.

IBM MQ Operator 1.7.0

CD

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2021.4.1

Canal opérateur
`v1.7`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.0-r3, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus, 9.2.0.2-r2-eus, 9.2.1.0-r1, 9.2.1.0-r2, 9.2.2.0-r1, 9.2.3.0-r1, 9.2.4.0-r1

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 et versions ultérieures

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services3.8 et version supérieure (canal v3)

Nouveautés

- Ajoute IBM MQ 9.2.4 comme version de prestation de services en continu

IBM MQ Operator 1.6.0

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2021.2.1

Canal opérateur
`v1.6`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.0-r3, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus, 9.2.0.2-r2-eus, 9.2.1.0-r1, 9.2.1.0-r2, 9.2.2.0-r1, 9.2.3.0-r1

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 et versions ultérieures

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services 3.7 et version supérieure (canal v3)

Nouveautés

- Ajoute IBM MQ 9.2.3 comme version de distribution continue (`amd64` uniquement pour IBM Cloud Pak for Integration 2021.2.1 ; `amd64` ou `s390x` lors de l'utilisation d'une licence IBM MQ)
- Nouveau type de disponibilité pour les gestionnaires de files d'attente : Native HA. Disponible pour une utilisation en production, dans le cadre de IBM Cloud Pak for Integration 2021.2.1.

Modifications

- IBM MQ Operator 1.6 et versions ultérieures utilisent IBM Container Registry au lieu de Docker Hub. Cela signifie que vous devez utiliser un `CatalogSource` à partir de `icr.io`. Voir «Installation et désinstallation de IBM MQ Operator sous Red Hat OpenShift», à la page 62.
- La mise à jour tournante de Native HA n'attend plus qu'un serveur secondaire soit synchronisé pour passer au serveur secondaire suivant.
- Correction du problème d'affinité Native HA sur OCP 4.7 et versions supérieures.
- Correction du problème lors de l'utilisation de certificats signés par une autorité de certification avec Native HA.

IBM MQ Operator 1.5.0

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2021.1.1

Canal opérateur
`v1.5`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.0-r3, 9.2.0.1-r1-eus, 9.2.1.0-r1, 9.2.1.0-r2, 9.2.2.0-r1

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 et versions ultérieures

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services 3.7 et version supérieure (canal v3)

Nouveautés

- Ajoute IBM MQ 9.2.2 comme version de distribution continue (amd64 uniquement pour IBM Cloud Pak for Integration 2021.1.1 ; amd64 ou s390x lors de l'utilisation d'une licence IBM MQ)
- Nouveau type de disponibilité pour les gestionnaires de files d'attente : Native HA. Disponible uniquement à des fins d'évaluation, dans le cadre d'IBM Cloud Pak for Integration 2021.1.1.
- Intégration avec Red Hat OpenShift Container Platform Cluster Monitoring pour les mesures de Prometheus, en fournissant une ressource `ServiceMonitor`

Modifications

- IBM Licensing Operator n'est plus créé par défaut lorsque vous créez un gestionnaire de files d'attente
- Les mises à jour des gestionnaires de files d'attente multi-instance sont désormais traitées par roulement. Dans le cadre de cette modification, une sonde de démarrage Kubernetes a été introduite qui affecte les valeurs utilisées lors de la configuration de la sonde de non-défaillance. La sonde de démarrage démarre immédiatement, puis attend que le gestionnaire de files d'attente démarre avec succès. Si la sonde de démarrage réussit à tout moment pendant cette période d'attente, les sondes de non-défaillance et de préparation démarrent alors. Auparavant, si vous aviez un gestionnaire de files d'attente qui était lent à démarrer, vous avez peut-être augmenté le paramètre `initialDelaySeconds` sur la sonde d'activité. Si vous l'avez fait, vous devez maintenant rétablir `initialDelaySeconds` dans le paramètre précédent.
- Le `CustomResourceDefinition` est mis à niveau de `apiextensions.k8s.io/v1beta1` vers `apiextensions.k8s.io/v1`

Problèmes connus et limitations

- Requiert IBM Cloud Pak foundational services 3.7, qui contient une modification non compatible dans le composant IAM (Identity and Access Management). Si vous disposez de gestionnaires de files d'attente qui utilisent une licence IBM Cloud Pak for Integration, après cette mise à niveau, un redémarrage du gestionnaire de files d'attente est requis pour accéder à la console Web, et vous observerez également d'autres erreurs de connexion à la console Web. Vous pouvez corriger ces erreurs en procédant à une mise à niveau vers la valeur la plus récente de `.spec.version` pour la version de IBM MQ de votre choix, une fois l'opérateur a été mis à niveau.
- La mise à jour par roulement ne démarre pas automatiquement si vous mettez à niveau la version MQ. Vous devez supprimer manuellement les pods.

IBM MQ Operator 1.4.0

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.4.1 (IBM MQ Operator 1.4.0 est une édition CD et n'est pas éligible pour l'édition EUS (Extended Update Support))

Canal opérateur
`v1.4`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.1-r1-eus, 9.2.1.0-r1

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 et versions ultérieures

Nouveautés

- Ajout d'IBM MQ 9.2.1 comme édition de distribution continue
- Vous pouvez maintenant empêcher la création de la route du gestionnaire de files d'attente par défaut en définissant `.spec.queueManager.route.enabled` sur `false`

Problèmes connus et limitations

- Lors de la mise à jour d'un `QueueManager` avec un type de disponibilité de `MultiInstance`, les deux Pods seront immédiatement supprimés. Ils doivent être redémarrés rapidement par Red Hat OpenShift Container Platform.

IBM MQ Operator 1.3.8 (EUS)

EUS

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.4.1

Canal opérateur
`v1.3-eus`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus, 9.2.0.4-r1-eus, 9.2.0.5-r1-eus, 9.2.0.5-r2-eus, 9.2.0.5-r3-eus, 9.2.0.6-r1-eus, 9.2.0.6-r2-eus, 9.2.0.6-r3-eus

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 uniquement

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services 3.6 (canal `stable-v1`)

Nouveautés

- Ajoute une nouvelle version d'opérande 9.2.0.6-r3-eus.
- Les vulnérabilités qui sont prises en compte sont détaillées dans ce bulletin de sécurité.

IBM MQ Operator 1.3.7 (EUS)

EUS

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.4.1

Canal opérateur
`v1.3-eus`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus, 9.2.0.4-r1-eus, 9.2.0.5-r1-eus, 9.2.0.5-r2-eus, 9.2.0.5-r3-eus, 9.2.0.6-r1-eus, 9.2.0.6-r2-eus

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 uniquement

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services 3.6 (canal `stable-v1`)

Nouveautés

- Ajoute une nouvelle version d'opérande 9.2.0.6-r2-eus.
- Les vulnérabilités qui sont prises en compte sont détaillées dans ce bulletin de sécurité.

IBM MQ Operator 1.3.6 (EUS)

EUS

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.4.1

Canal opérateur
`v1.3-eus`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus, 9.2.0.4-r1-eus, 9.2.0.5-r1-eus, 9.2.0.5-r2-eus, 9.2.0.5-r3-eus, 9.2.0.6-r1-eus

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 uniquement

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services 3.6 (canal `stable-v1`)

Nouveautés

- Ajoute une nouvelle version d'opérande 9.2.0.6-r1-eus.
- Les vulnérabilités qui sont prises en compte sont détaillées dans ce bulletin de sécurité.

IBM MQ Operator 1.3.5 (EUS)

EUS

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.4.1

Canal opérateur
`v1.3-eus`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus, 9.2.0.4-r1-eus, 9.2.0.5-r1-eus, 9.2.0.5-r2-eus, 9.2.0.5-r3-eus

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 uniquement

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services 3.6 (canal `stable-v1`)

Nouveautés

- Ajoute une nouvelle version d'opérande 9.2.0.5-r3-eus.
- Les vulnérabilités qui sont prises en compte sont détaillées dans ce bulletin de sécurité.

IBM MQ Operator 1.3.4 (EUS)

EUS

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.4.1

Canal opérateur
`v1.3-eus`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus, 9.2.0.4-r1-eus, 9.2.0.5-r1-eus, 9.2.0.5-r2-eus

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 uniquement

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services 3.6 (canal `stable-v1`)

Nouveautés

- Ajout d'une nouvelle version d'opérande 9.2.0.5-r2-eus
- Les vulnérabilités qui sont prises en compte sont détaillées dans ce bulletin de sécurité.

IBM MQ Operator 1.3.3 (EUS)

EUS

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.4.1

Canal opérateur
`v1.3-eus`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus, 9.2.0.4-r1-eus, 9.2.0.5-r1-eus

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 uniquement

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services 3.6 (canal `stable-v1`)

Nouveautés

- Ajoute une nouvelle version d'opérande 9.2.0.5-r1-eus
- Les vulnérabilités qui sont prises en compte sont détaillées dans ce bulletin de sécurité.

IBM MQ Operator 1.3.2 (EUS)

EUS

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.4.1

Canal opérateur
`v1.3-eus`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus, 9.2.0.4-r1-eus

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 uniquement

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services 3.6 (canal `stable-v1`)

Nouveautés

- Ajoute une nouvelle version du facteur 9.2.0.4-r1-eus

IBM MQ Operator 1.3.1 (EUS)

EUS

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.4.1

Canal opérateur
`v1.3-eus`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.1-r1-eus, 9.2.0.2-r1-eus

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 uniquement

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services 3.6 (canal `stable-v1`)

Nouveautés

- Ajout d'une nouvelle version d'opérande 9.2.0.2-r1-eus

IBM MQ Operator 1.3.0 (EUS)

EUS

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.4.1

Canal opérateur
`v1.3-eus`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.1-r1-eus

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.6 uniquement

Versions IBM Cloud Pak foundational services
IBM Cloud Pak foundational services 3.6 (canal `stable-v1`)

Nouveautés

- Extended Update Support (EUS) est proposé pour les zones `.spec.version` se terminant par `-eus`, lors de l'utilisation d'une licence IBM Cloud Pak for Integration
- Ajoute une nouvelle façon de définir des libellés et des annotations sur la ressource `QueueManager` à l'aide de `.spec.labels` et de `.spec.annotations`

Modifications

- Amélioration de la gestion des erreurs lors de la tentative de modification d'une instance unique à une multi-instance
- Amélioration de la manière dont les propriétés `QueueManager` sont rendues dans IBM Cloud Pak for Integration Platform Navigator, et la "Vue du formulaire" de la console Web Red Hat OpenShift Container Platform
- Corrige la mesure de licence par défaut lors de l'utilisation d'une licence IBM Cloud Pak for Integration, pour être `VirtualProcessorCore`
- Corrige l'onglet **Ressources** pour `QueueManager` dans la console Web Red Hat OpenShift Container Platform, qui affiche désormais correctement les ressources gérées par IBM MQ Operator pour ce gestionnaire de files d'attente

Problèmes connus et limitations

- Lors de la mise à jour d'un `QueueManager` avec un type de disponibilité de `MultiInstance`, les deux Pods seront immédiatement supprimés. Ils doivent être redémarrés rapidement par Red Hat OpenShift Container Platform.

IBM MQ Operator 1.2.0

CD

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.3.1

Canal opérateur
`v1.2`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.4 et versions ultérieures

Nouveautés

- Ajout d'une prise en charge pour z/Linux
- Ajoute des conditions d'état plus détaillées à la ressource `QueueManager`. Pour plus d'informations, voir «Conditions de statut de QueueManager (mq.ibm.com/v1beta1)», à la page 150
- Ajout de vérifications d'exécution pour empêcher l'utilisation de classes de stockage non valides. Pour plus d'informations, voir «Désactivation des vérifications des webhooks d'exécution», à la page 119

- Simplifie l'expérience pour les gestionnaires de files d'attente multi-instance : cela peut désormais être choisi avec une seule propriété (`.spec.queueManager.availability.type`) dans la ressource `QueueManager`
- Simplifie le choix d'une classe de stockage autre que celle par défaut, en introduisant la propriété `.spec.queueManager.storage.defaultClass` dans la ressource `QueueManager`

Modifications

- Amélioration de la manière dont les propriétés `QueueManager` sont rendues dans IBM Cloud Pak for Integration Platform Navigator, et la "Vue du formulaire" de la console Web Red Hat OpenShift Container Platform
- Si une version de gestionnaire de files d'attente mise à niveau est disponible, elle est désormais marquée dans IBM Cloud Pak for Integration Platform Navigator

IBM MQ Operator 1.1.0

CD

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.2.1

Canal opérateur
`v1.1`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.4 et versions ultérieures

Nouveautés

- Ajout d'IBM MQ Advanced 9.2.0 comme édition de distribution continue
- Ajout d'une fonction pour spécifier les informations INI et MQSC dans une mappe de configuration ou un secret
- Activation du navigateur de schémas lors de l'utilisation de la console Web Red Hat OpenShift Container Platform

Modifications

- Corrige un problème avec la stratégie de réseau, ce qui a un impact sur Red Hat OpenShift sous IBM Cloud
- Améliorations apportées au point d'ancrage Web de validation pour empêcher les combinaisons de paramètres non valides dans les ressources `QueueManager`

IBM MQ Operator 1.0.0

CD

Version de IBM Cloud Pak for Integration
IBM Cloud Pak for Integration 2020.2.1

Canal opérateur
`v1.0`

Valeurs autorisées pour `.spec.version`
9.1.5.0-r2

Versions Red Hat OpenShift Container Platform
Red Hat OpenShift Container Platform 4.4 et versions ultérieures

Nouveautés

- Version initiale de l'opérateur, introduisant l'API `mq.ibm.com/v1beta1`

Images de conteneur de gestionnaire de files d'attente à utiliser avec le IBM MQ Operator

9.2.5.0-r3

CD

Version de l'opérateur requise
1.8.2 ou version ultérieure

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.5.0-r3`
- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.5.0-r3`
- `icr.io/ibm-messaging/mq:9.2.5.0-r3`

Nouveautés

- Nouveautés de IBM MQ 9.2.5

Modifications

- Ce qui a changé dans IBM MQ 9.2.5
- Basé sur Red Hat Universal Base Image 8.6-751

9.2.5.0-r2

CD

Version de l'opérateur requise
1.8.1 ou version ultérieure

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.5.0-r2`
- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.5.0-r2`
- `icr.io/ibm-messaging/mq:9.2.5.0-r2`

Nouveautés

- Nouveautés de IBM MQ 9.2.5

Modifications

- Ce qui a changé dans IBM MQ 9.2.5
- Basé sur Red Hat Universal Base Image 8.5-240.1648458092

9.2.5.0-r1

CD

Version de l'opérateur requise
1.8.0 ou version ultérieure

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.5.0-r1`

- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.5.0-r1`
- `icr.io/ibm-messaging/mq:9.2.5.0-r1`

Nouveautés

- Nouveautés de IBM MQ 9.2.5

Modifications

- Ce qui a changé dans IBM MQ 9.2.5
- Option `Gestionnaires de files d'attente éloignées` non valide maintenant supprimée de IBM MQ Console
- Basé sur Red Hat Universal Base Image 8.5-240

9.2.4.0-r1

Version de l'opérateur requise
1.7.0 ou version ultérieure

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.4.0-r1`
- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.4.0-r1`
- `docker.io/ibmcom/mq:9.2.4.0-r1`

Nouveautés

- Nouveautés de IBM MQ 9.2.4

Modifications

- Ce qui a changé dans IBM MQ 9.2.4
- Basé sur Red Hat Universal Base Image 8.5-204

9.2.3.0-r1

Version de l'opérateur requise
1.6.0 ou versions ultérieures

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.3.0-r1` (amd64 uniquement)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.3.0-r1`
- `docker.io/ibmcom/mq:9.2.3.0-r1`

Nouveautés

- Nouveautés de IBM MQ 9.2.3
- Prise en charge de MQ Native HA pour une utilisation en production, lorsqu'il est utilisé avec une licence IBM Cloud Pak for Integration. Notez que les gestionnaires de files d'attente utilisant Native HA sous une licence d'évaluation avec IBM MQ 9.2.2 ne peuvent pas être mis à niveau vers 9.2.3. La période d'évaluation est terminée.

Modifications

- Ce qui a changé dans IBM MQ 9.2.3

- Basé sur Red Hat Universal Base Image 8.4-205

9.2.2.0-r1

CD

Version de l'opérateur requise
1.5.0 ou version ultérieure

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.2.0-r1` (amd64 uniquement)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.2.0-r1`
- `docker.io/ibmcom/mq:9.2.2.0-r1`

Nouveautés

- Nouveautés de IBM MQ 9.2.2
- Prise en charge de la fonctionnalité Native HA de MQ à des fins d'évaluation si elle est utilisée avec une licence IBM Cloud Pak for Integration

Modifications

- Ce qui a changé dans IBM MQ 9.2.2
- Résolution d'un problème à l'origine de FDC lors de l'arrêt d'un gestionnaire de files d'attente IBM MQ Advanced for Developers
- Basé sur Red Hat Universal Base Image 8.3-291

9.2.1.0-r2

CD

Version de l'opérateur requise
1.5.0 ou version ultérieure

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.1.0-r2`
- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.1.0-r2`
- `docker.io/ibmcom/mq:9.2.1.0-r2`

Modifications

- Correction du problème de connexion unique avec IBM Cloud Pak foundational services 3.7 et versions supérieures.
- Basé sur Red Hat Universal Base Image 8.3-291

9.2.1.0-r1

CD

Version de l'opérateur requise
1.4.0 ou version ultérieure

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.1.0-r1`
- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.1.0-r1`
- `docker.io/ibmcom/mq:9.2.1.0-r1`

Nouveautés

- Nouveautés de IBM MQ 9.2.1
- Les informations de connexion pour la route par défaut sont disponibles dans la console Web MQ.

Modifications

- Ce qui a changé dans IBM MQ 9.2.1
- Basé sur Red Hat Universal Base Image 8.3-230

9.2.0.6-r3-eus

EUS

Version de l'opérateur requise
1.3.8 et groupes de correctifs ultérieurs

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.6-r3-eus`

Modifications

- Inclut IBM MQ 9.2.0 Fix Pack 6. Pour plus d'informations, voir Fix list for IBM MQ Version 9.2 LTS.
- Basé sur Red Hat Universal Base Image 8.6-941.

9.2.0.6-r2-eus

EUS

Version de l'opérateur requise
1.3.7 et groupes de correctifs ultérieurs

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.6-r2-eus`

Modifications

- Inclut IBM MQ 9.2.0 Fix Pack 6. Pour plus d'informations, voir Fix list for IBM MQ Version 9.2 LTS.
- Basé sur Red Hat Universal Base Image 8.6-902.

9.2.0.6-r1-eus

EUS

Version de l'opérateur requise
1.3.6 et groupes de correctifs ultérieurs

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.6-r1-eus`

Modifications

- Inclut IBM MQ 9.2.0 Fix Pack 6. Pour plus d'informations, voir Fix list for IBM MQ Version 9.2 LTS.
- Basé sur Red Hat Universal Base Image 8.6-854.

9.2.0.5-r3-eus

EUS

Version de l'opérateur requise
1.3.5 et groupes de correctifs ultérieurs

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.5-r3-eus`

Modifications

- Inclut IBM MQ 9.2.0 Fix Pack 5. Pour plus d'informations, voir What's changed in IBM MQ 9.2.0 Fix Pack 5 and Fix list for IBM MQ Version 9.2 LTS.
- Basé sur Red Hat Universal Base Image 8.6-751.1655117800.

9.2.0.5-r2-eus

EUS

Version de l'opérateur requise
1.3.4 et groupes de correctifs ultérieurs

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.5-r2-eus`

Modifications

- Inclut IBM MQ 9.2.0 Fix Pack 5. Pour plus d'informations, voir Nouveautés de IBM MQ 9.2.0 Fix Pack 5 et Liste de correctifs pour IBM MQ version 9.2 LTS
- Basé sur Red Hat Universal Base Image 8.6-751

9.2.0.5-r1-eus

EUS

Version de l'opérateur requise
1.3.3 et groupes de correctifs ultérieurs

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.5-r1-eus`

Modifications

- Inclut IBM MQ 9.2.0 Fix Pack 5. Pour plus d'informations, voir Nouveautés de IBM MQ 9.2.0 Fix Pack 5 et Liste de correctifs pour IBM MQ version 9.2 LTS
- Basé sur Red Hat Universal Base Image 8.5-240.1648458092

9.2.0.4-r1-eus

EUS

Version de l'opérateur requise
1.3.2 et groupes de correctifs ultérieurs

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.4-r1-eus`

Modifications

- Inclut IBM MQ 9.2.0 Fix Pack 4. Pour plus d'informations, voir Nouveautés de IBM MQ 9.2.0 Fix Pack 4 et Liste de correctifs pour IBM MQ version 9.2 LTS
- Basé sur Red Hat Universal Base Image 8.5-204

9.2.0.2-r2-eus

EUS

Version de l'opérateur requise
1.6.0 ou versions ultérieures

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.2-r2-eus`

Modifications

- Correction du problème de connexion unique avec IBM Cloud Pak foundational services 3.7 et versions supérieures, qui n'est nécessaire que lors de la migration d'une édition EUS vers une édition CD.
- Basé sur Red Hat Universal Base Image 8.4-200.1622548483

9.2.0.2-r1-eus

EUS

Version de l'opérateur requise
1.3.1 et groupes de correctifs ultérieurs ; 1.6.0 ou supérieure

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.2-r1-eus`

Modifications

- L'intégration du tableau de bord des opérations utilise l'agent de traçage et le collecteur version 1.0.8
- Inclut IBM MQ 9.2.0 Fix Pack 2. Pour plus d'informations, voir Nouveautés de IBM MQ 9.2.0 Fix Pack 2 et Liste de correctifs pour IBM MQ version 9.2 LTS
- Basé sur Red Hat Universal Base Image 8.4-200.1622548483

9.2.0.1-r1-eus

EUS

Version de l'opérateur requise
1.3.0 ou version ultérieure

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.1-r1-eus`

Nouveautés

- Uniquement disponible lors de l'utilisation d'une licence IBM Cloud Pak for Integration
- L'édition EUS (Extended Update Support) est disponible lors de l'utilisation de IBM MQ Operator 1.3.x et d'IBM Common Services 3.6, sous Red Hat OpenShift Container Platform 4.6

Modifications

- Inclut IBM MQ 9.2.0 Fix Pack 1. Pour plus d'informations, voir Nouveautés de IBM MQ 9.2.0 Fix Pack 1 et Liste de correctifs pour IBM MQ version 9.2 LTS
- Basé sur Red Hat Universal Base Image 8.3-201
- Fixe un problème avec la sonde d'activité (`chkmqhealthy`) et la sonde de conformité (`chkmqready`) lors de l'exécution sous `SecurityContextContraints`, ce qui permet une escalade des privilèges.

9.2.0.0-r3

CD

Version de l'opérateur requise
1.5.0 ou version ultérieure

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.0-r3`
- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.0.0-r3`
- `docker.io/ibmcom/mq:9.2.0.0-r3`

Modifications

- Basé sur Red Hat Universal Base Image 8.3-291

9.2.0.0-r2

CD

Version de l'opérateur requise
1.2.0 ou version ultérieure

Architectures prises en charge
amd64, s390x

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.0-r2`
- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.0.0-r2`
- `docker.io/ibmcom/mq:9.2.0.0-r2`

Nouveautés

- Maintenant disponible sur z/Linux

Modifications

- Basé sur Red Hat Universal Base Image 8.2-349

9.2.0.0-r1

CD

Version de l'opérateur requise
1.1.0 ou version ultérieure

Architectures prises en charge
amd64

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.0.0-r1-amd64`
- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.0.0-r1-amd64`
- `docker.io/ibmcom/mq:9.2.0.0-r1`

Nouveautés

- Nouveautés de IBM MQ 9.2.0

Modifications

- Ce qui a changé dans IBM MQ 9.2.0
- Utilise l'argument `-ic` pour `crtmqm` afin d'appliquer automatiquement les fichiers MQSC. Remplace l'utilisation précédente des commandes `runmqsc`
- Basé sur Red Hat Universal Base Image 8.2-301.1593113563

9.1.5.0-r2

CD

Version de l'opérateur requise
1.0.0 ou version ultérieure

Architectures prises en charge
amd64

Images

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.1.5.0-r2-amd64`
- `cp.icr.io/cp/ibm-mqadvanced-server:9.1.5.0-r2-amd64`
- `docker.io/ibmcom/mq:9.1.5.0-r2`

Modifications

- Basé sur Red Hat Universal Base Image 8.2-267

OpenShift CD V 9.2.1 EUS Migration d'IBM MQ vers IBM Cloud Pak for Integration

Cet ensemble de rubriques décrit les principales étapes de la migration d'un gestionnaire de files d'attente IBM MQ existant vers un environnement de conteneur à l'aide de IBM MQ Operator dans IBM Cloud Pak for Integration.

Pourquoi et quand exécuter cette tâche

Les clients qui déploient IBM MQ sous Red Hat OpenShift peuvent être séparés dans les scénarios suivants :

1. Création d'un déploiement IBM MQ dans Red Hat OpenShift pour les nouvelles applications.

2. Extension d'un réseau IBM MQ dans Red Hat OpenShift pour les nouvelles applications dans Red Hat OpenShift.
3. Transfert d'un déploiement IBM MQ dans Red Hat OpenShift pour continuer à prendre en charge les applications existantes.

Seul le scénario 3 requiert que vous migriez votre configuration IBM MQ. Les autres scénarios sont considérés comme de nouveaux déploiements.

Cet ensemble de rubriques se concentre sur le scénario 3 et décrit les principales étapes de la migration d'un gestionnaire de files d'attente IBM MQ existant dans un environnement de conteneur à l'aide de IBM MQ Operator. En raison de la flexibilité et de l'utilisation intensive d'IBM MQ, il existe plusieurs étapes facultatives. Chacune d'elles inclut une section "Cette tâche est-elle obligatoire ?". Identifiez vos besoins pour gagner du temps lors de la migration.

Vous devez également déterminer les données à migrer :

1. Migrer IBM MQ avec la même configuration, mais sans les messages de file d'attente existants.
2. Migrer IBM MQ avec la même configuration et les messages existants.

Une migration type de version à version peut utiliser l'une ou l'autre de ces approches. Dans un gestionnaire de files d'attente IBM MQ type au point de migration, seuls quelques messages sont éventuellement stockés dans des files d'attente, ce qui rend l'option 1 appropriée dans de nombreux cas. Dans le cas d'une migration vers une plateforme de conteneur, il est encore plus courant d'utiliser l'option 1 pour réduire la complexité de la migration et permettre un déploiement Blue Green. Par conséquent, les instructions portent sur ce scénario.

Ce scénario a pour objectif de créer un gestionnaire de files d'attente dans l'environnement de conteneur qui correspond à la définition du gestionnaire de files d'attente existant. Ainsi, les applications existantes connectées au réseau peuvent simplement être reconfigurées pour pointer vers le nouveau gestionnaire de files d'attente, sans qu'il ne soit nécessaire de modifier le reste de la configuration ou la logique d'application.

Tout au long de cette migration, vous générez plusieurs fichiers de configuration à appliquer au nouveau gestionnaire de files d'attente. Pour simplifier la gestion de ces fichiers, vous devez créer un répertoire et les générer dans ce répertoire.

Procédure

1. «Vérification de la disponibilité des fonctions requises», à la page 39
2. «Extraction de la configuration du gestionnaire de files d'attente», à la page 39
3. Facultatif : «Facultatif : extraction et acquisition des clés et certificats du gestionnaire de files d'attente», à la page 40
4. Facultatif : «Facultatif : configuration de LDAP», à la page 42
5. Facultatif : «Facultatif : modification des adresses IP et noms d'hôte dans la configuration IBM MQ», à la page 50
6. «Mise à jour de la configuration du gestionnaire de files d'attente pour un environnement de conteneur», à la page 51
7. «Sélection de l'architecture haute disponibilité cible pour IBM MQ exécuté en conteneurs», à la page 55
8. «Création des ressources du gestionnaire de files d'attente», à la page 55
9. «Création du gestionnaire de files d'attente dans Red Hat OpenShift», à la page 56
10. «Vérification du nouveau déploiement de conteneur», à la page 60

OpenShift CD V 9.2.1 EUS Vérification de la disponibilité des fonctions requises

Le IBM MQ Operator n'inclut pas toutes les fonctionnalités disponibles dans IBM MQ Advanced et vous devez vérifier que ces dernières ne sont pas requises. D'autres fonctionnalités sont partiellement prises en charge et peuvent être reconfigurées conformément à celles disponibles dans le conteneur.

Avant de commencer

Il s'agit de la première étape de la rubrique «Migration d'IBM MQ vers IBM Cloud Pak for Integration», à la page 37.

Procédure

1. Vérifiez que l'image du conteneur cible inclut toutes les fonctions requises.

   Pour les informations les plus récentes, reportez-vous à la rubrique «Comment utiliser IBM MQ dans des conteneurs», à la page 5.
2. Le IBM MQ Operator possède un unique port de trafic IBM MQ, appelé programme d'écoute. Si vous disposez de plusieurs programmes d'écoute, procéder à une simplification pour n'utiliser qu'un seul programme d'écoute dans le conteneur. Comme il ne s'agit pas d'un scénario courant, cette modification n'est pas décrite en détail.
3. Si des exits IBM MQ sont utilisés, migrez-les dans le conteneur en les superposant dans les fichiers binaires d'exit de IBM MQ. Il s'agit d'un scénario de migration avancé, qui n'est donc pas inclus ici. Pour un aperçu des étapes, reportez-vous à la rubrique «Génération d'une image avec des fichiers MQSC et INI personnalisés, à l'aide de l'interface de ligne de commande Red Hat OpenShift», à la page 117.
4. Si votre système IBM MQ inclut la haute disponibilité, vérifiez les options disponibles.

   Voir «Haute disponibilité pour IBM MQ dans les conteneurs», à la page 16.

Que faire ensuite

Vous êtes maintenant prêt à extraire la configuration du gestionnaire de files d'attente.

OpenShift CD V 9.2.1 EUS Extraction de la configuration du gestionnaire de files d'attente

La majorité de la configuration est transférable entre les gestionnaires de files d'attente, notamment les éléments avec lesquels les applications interagissent, comme les définitions des files d'attente, des rubriques et des canaux. Utilisez cette tâche pour extraire la configuration du gestionnaire de files d'attente IBM MQ existant.

Avant de commencer

Cette tâche suppose que vous avez vérifié que les fonctions requises sont disponibles.

Procédure

1. Connectez-vous à la machine sur laquelle IBM MQ est installé.
2. Sauvegardez la configuration.

   Exécutez ensuite la commande suivante :

   ```
   dmpmqcfg -m QMGR_NAME > /tmp/backup.mqsc
   ```

   Remarques sur l'utilisation de cette commande :

   - Cette commande stocke la sauvegarde dans le répertoire `tmp`. Vous pouvez la stocker dans un autre emplacement, mais le présent scénario utilise le répertoire `tmp` pour les commandes suivantes.

- Remplacez *QMGR_NAME* par le nom du gestionnaire de files d'attente de votre environnement. Si vous n'êtes pas sûr de la valeur, exécutez la commande **dspmq** pour afficher les gestionnaires de files d'attente disponibles sur la machine. Vous trouverez ci-dessous un exemple de sortie de commande **dspmq** pour un gestionnaire de files d'attente nommé qm1 :

```
QMNAME(qm1)                   STATUS(Running)
```

La commande **dspmq** requiert que le gestionnaire de files d'attente IBM MQ soit démarré, faute de quoi l'erreur suivante est générée :

```
AMQ8146E: IBM MQ queue manager not available.
```

Si nécessaire, démarrez le gestionnaire de files d'attente à l'aide de la commande suivante :

```
strmqm QMGR_NAME
```

Que faire ensuite

Vous êtes maintenant prêt à extraire et acquérir les clés et certificats du gestionnaire de files d'attente.

OpenShift CD V 9.2.1 EUS Facultatif : extraction et acquisition des clés et certificats du gestionnaire de files d'attente

IBM MQ peut être configuré à l'aide de TLS pour chiffrer le trafic dans le gestionnaire de files d'attente. Utilisez cette tâche pour vérifier si votre gestionnaire de files d'attente utilise TLS, extraire des clés et des certificats et configurer TLS sur le gestionnaire de files d'attente migré.

Avant de commencer

Cette tâche suppose que vous avez extrait la configuration du gestionnaire de files d'attente.

Pourquoi et quand exécuter cette tâche

Cette tâche est-elle obligatoire ?

IBM MQ peut être configuré pour chiffrer le trafic dans le gestionnaire de files d'attente. Ce chiffrement est réalisé à l'aide d'un référentiel de clés configuré sur le gestionnaire de files d'attente. Les canaux IBM MQ permettent ensuite les communications TLS. Si vous n'êtes pas certain qu'il soit configuré dans votre environnement, exécutez la commande suivante pour le vérifier :

```
grep 'SECCOMM(ALL\|SECCOMM(ANON\|SSLCIPH' backup.mqsc
```

Si aucun résultat n'est trouvé, TLS n'est pas utilisé. Toutefois, cela ne signifie pas que TLS ne doit pas être configuré dans le gestionnaire de files d'attente migré. Il existe plusieurs raisons pour lesquelles vous pouvez modifier ce comportement :

- L'approche de sécurité de l'environnement Red Hat OpenShift doit être améliorée par rapport à l'environnement précédent.
- Si vous devez accéder au gestionnaire de files d'attente migré à partir de l'extérieur de l'environnement Red Hat OpenShift, TLS doit passer par la route Red Hat OpenShift.

Procédure

1. Extrayez les certificats sécurisés du magasin existant.

   Si TLS est actuellement utilisé dans le gestionnaire de files d'attente, un certain nombre de certificats sécurisés sont sans doute stockés pour ce dernier. Ils doivent être extraits et copiés dans le nouveau gestionnaire de files d'attente. Effectuez l'une des étapes facultatives suivantes :

- Pour rationaliser l'extraction des certificats, exécutez le script suivant sur le système local :

```
#!/bin/bash

keyr=$(grep SSLKEYR $1)
if [ -n "${keyr}" ]; then
    keyrlocation=$(sed -n "s/^.*'\(.*\)'.*$/\1/ p" <<< ${keyr})
    mapfile -t runmqckmResult < <(runmqckm -cert -list -db ${keyrlocation}.kdb -stashed)
        cert=1
    for i in "${runmqckmResult[@]:1}"
        do
           certlabel=$(echo ${i} | xargs)
           echo Extracting certificate $certlabel to $cert.cert
       runmqckm -cert -extract -db ${keyrlocation}.kdb -label "$certlabel" -target $
{cert}.cert -stashed
       cert=$[$cert+1]
        done
fi
```

  Lors de l'exécution de ce script, spécifiez l'emplacement de la sauvegarde IBM MQ comme argument et les certificats sont extraits. Par exemple, si le script s'intitule `extractCert.sh` et que la sauvegarde IBM MQ se trouve dans `/tmp/backup.mqsc`, exécutez la commande suivante :

```
extractCert.sh /tmp/backup.mqsc
```

- Vous pouvez également exécuter les commandes suivantes suivant l'ordre indiqué :

  a. Identifiez l'emplacement du magasin TLS :

  ```
  grep SSLKEYR /tmp/backup.mqsc
  ```

  Exemple de sortie :

  ```
  SSLKEYR('/run/runmqserver/tls/key') +
  ```

  Où le magasin de clés se trouve dans `/run/runmqserver/tls/key.kdb`

  b. En fonction de ces informations d'emplacement, interrogez le magasin de clés pour déterminer les certificats stockés :

  ```
  runmqckm -cert -list -db /run/runmqserver/tls/key.kdb -stashed
  ```

  Exemple de sortie :

  ```
  Certificates in database /run/runmqserver/tls/key.kdb:
            default
            CN=cs-ca-certificate,O=cert-manager
  ```

  c. Extrayez chacun des certificats répertoriés, à l'aide de la commande suivante :

  ```
  runmqckm -cert -extract -db KEYSTORE_LOCATION -label "LABEL_NAME" -target OUTPUT_FILE
  -stashed
  ```

  Dans les exemples précédemment affichés, cela correspond à ce qui suit :

  ```
  runmqckm -cert -extract -db /run/runmqserver/tls/key.kdb -label "CN=cs-ca-
  certificate,O=cert-manager" -target /tmp/cert-manager.crt -stashed
  runmqckm -cert -extract -db /run/runmqserver/tls/key.kdb -label "default" -target /tmp/
  default.crt -stashed
  ```

2. Procurez-vous une nouvelle clé et un nouveau certificat pour le gestionnaire de files d'attente.

   Pour configurer TLS sur le gestionnaire de files d'attente migré, générez une nouvelle clé et un nouveau certificat. Ces derniers seront utilisés lors du déploiement. Dans de nombreuses organisations, cela

implique de contacter votre équipe de sécurité pour demander une clé et un certificat. Dans certaines organisations, cette option n'est pas disponible et des certificats autosignés sont utilisés.

L'exemple suivant génère un certificat autosigné dont le délai d'expiration est défini sur 10 ans :

```
openssl req \
       -newkey rsa:2048 -nodes -keyout qmgr.key \
       -subj "/CN=mq queuemanager/OU=ibm mq" \
       -x509 -days 3650 -out qmgr.crt
```

Deux fichiers sont créés :

- `qmgr.key` est la clé privée du gestionnaire de files d'attente
- `qmgr.crt` est le certificat public

Que faire ensuite

Vous êtes maintenant prêt à configurer LDAP.

OpenShift CD V 9.2.1 EUS Facultatif : configuration de LDAP

Le IBM MQ Operator peut être configuré de sorte à utiliser plusieurs approches de sécurité différentes. En général, LDAP est le plus efficace pour un déploiement d'entreprise ; il est utilisé pour ce scénario de migration.

Avant de commencer

Cette tâche suppose que vous avez extrait et acquis les clés et certificats du gestionnaire de files d'attente.

Pourquoi et quand exécuter cette tâche

Cette tâche est-elle obligatoire ?

Si vous utilisez déjà LDAP pour l'authentification et l'autorisation, aucune modification n'est requise.

Si vous n'êtes pas certain que LDAP est utilisé, exécutez la commande suivante :

```
connauthname="$(grep CONNAUTH backup.mqsc | cut -d "(" -f2 | cut -d ")" -f1)"; grep -A 20
AUTHINFO\($connauthname\) backup.mqsc
```

Exemple de sortie :

```
DEFINE AUTHINFO('USE.LDAP') +
   AUTHTYPE(IDPWLDAP) +
   ADOPTCTX(YES) +
   CONNAME('ldap-service.ldap(389)') +
   CHCKCLNT(REQUIRED) +
   CLASSGRP('groupOfUniqueNames') +
   FINDGRP('uniqueMember') +
   BASEDNG('ou=groups,dc=ibm,dc=com') +
   BASEDNU('ou=people,dc=ibm,dc=com') +
   LDAPUSER('cn=admin,dc=ibm,dc=com') +
*  LDAPPWD('******************************') +
   SHORTUSR('uid') +
   GRPFIELD('cn') +
   USRFIELD('uid') +
   AUTHORMD(SEARCHGRP) +
*  ALTDATE(2020-11-26) +
*  ALTTIME(15.44.38) +
   REPLACE
```

Deux attributs de la sortie présentent un intérêt particulier :

AUTHTYPE
Si sa valeur est `IDPWLDAP`, vous utilisez LDAP pour l'authentification.

Si sa valeur est vide ou autre, LDAP n'est pas configuré. Dans ce cas, vérifiez l'attribut AUTHORMD pour déterminer si des utilisateurs LDAP sont utilisés pour l'autorisation.

AUTHORMD
Si sa valeur est OS, vous utilisez LDAP pour l'autorisation.

Pour modifier l'autorisation et l'authentification afin qu'elles utilisent LDAP, procédez comme suit :

Procédure

1. Mettez à jour la sauvegarde IBM MQ pour le serveur LDAP.
2. Mettez à jour la sauvegarde IBM MQ pour les informations d'autorisation LDAP.

LDAP - Partie 1 : mise à jour de la sauvegarde IBM MQ pour le serveur LDAP.

Ce scénario n'inclut pas de description complète de la procédure de configuration de LDAP. Cette rubrique résume la procédure et fournit un exemple et des références à des informations supplémentaires.

Avant de commencer

Cette tâche suppose que vous avez extrait et acquis les clés et certificats du gestionnaire de files d'attente.

Pourquoi et quand exécuter cette tâche

Cette tâche est-elle obligatoire ?

Si vous utilisez déjà LDAP pour l'authentification et l'autorisation, aucune modification n'est requise. Si vous n'êtes pas certain que LDAP est utilisé, reportez-vous à la rubrique «Facultatif : configuration de LDAP», à la page 42.

La configuration du serveur LDAP se déroule en deux parties :

1. Définissez une configuration LDAP.
2. Associez la configuration LDAP à la définition de gestionnaire de files d'attente.

Informations supplémentaires pour vous aider dans cette configuration :

- Présentation du référentiel d'utilisateurs
- Guide de référence de la commande AUTHINFO

Procédure

1. Définissez une configuration LDAP.

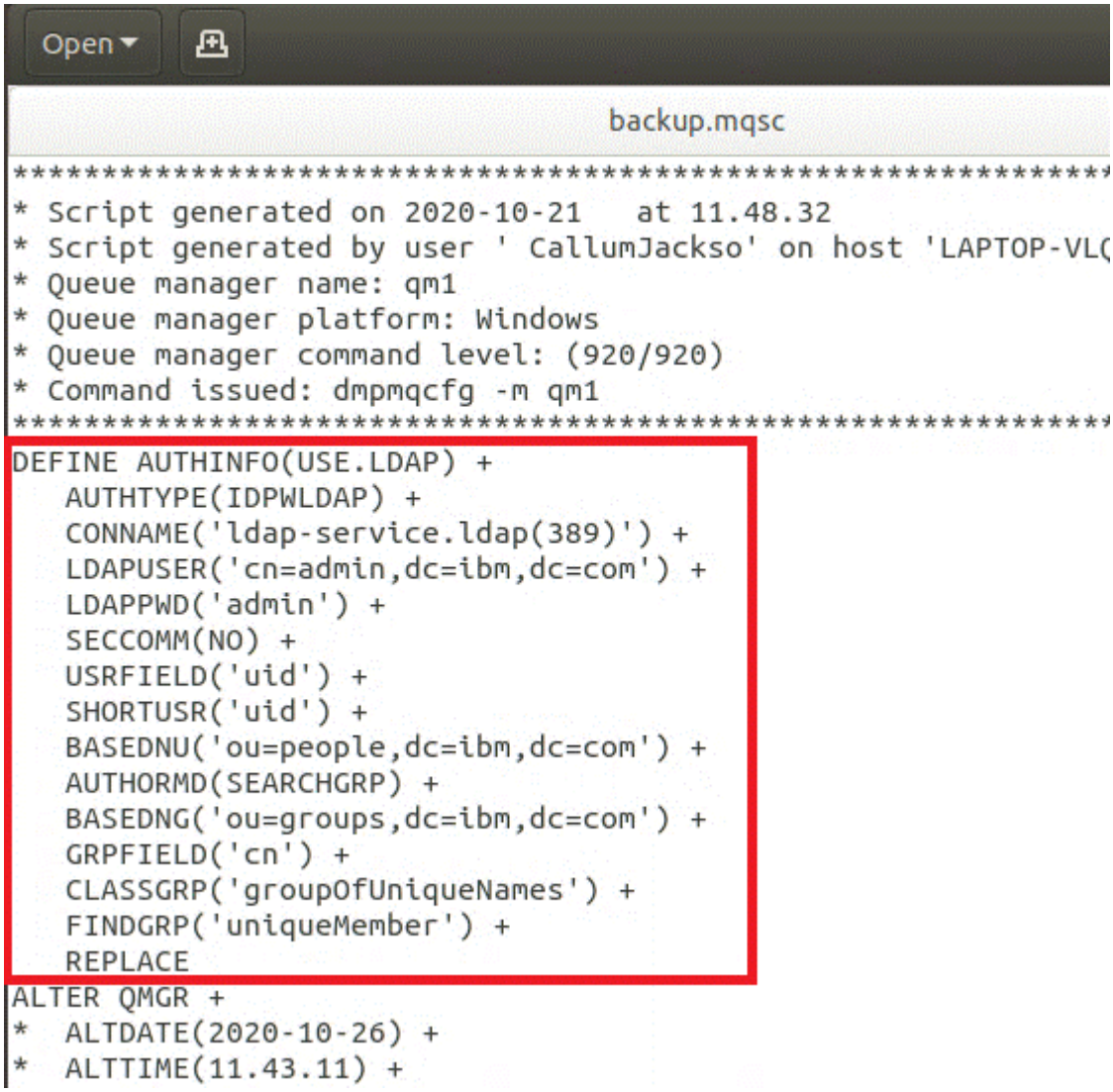
   Editez le fichier `backup.mqsc` afin de définir un nouvel objet **AUTHINFO** pour le système LDAP. Exemple :

```
DEFINE AUTHINFO(USE.LDAP) +
   AUTHTYPE(IDPWLDAP) +
   CONNAME('ldap-service.ldap(389)') +
   LDAPUSER('cn=admin,dc=ibm,dc=com') +
   LDAPPWD('admin') +
   SECCOMM(NO) +
   USRFIELD('uid') +
   SHORTUSR('uid') +
   BASEDNU('ou=people,dc=ibm,dc=com') +
   AUTHORMD(SEARCHGRP) +
   BASEDNG('ou=groups,dc=ibm,dc=com') +
   GRPFIELD('cn') +
   CLASSGRP('groupOfUniqueNames') +
   FINDGRP('uniqueMember')
   REPLACE
```

   Où

- **CONNAME** représente le nom d'hôte et le port correspondant au serveur LDAP. S'il existe plusieurs adresses à des fins de résilience, elles peuvent être configurées à l'aide d'une liste de valeurs séparées par des virgules.
- **LDAPUSER** représente le nom distinctif correspondant à l'utilisateur utilisé par IBM MQ lors de la connexion à LDAP pour interroger les enregistrements utilisateur.
- **LDAPPWD** représente le mot de passe qui correspond à l'utilisateur **LDAPUSER**.
- **SECCOM** indique si les communications avec le serveur LDAP doivent utiliser TLS. Valeurs possibles :
  - YES : TLS est utilisé et un certificat est présenté par le serveur IBM MQ.
  - ANON : TLS est utilisé sans certificat présenté par le serveur IBM MQ.
  - NO : TLS n'est pas utilisé lors de la connexion.
- **USRFIELD** spécifie la zone de l'enregistrement LDAP à laquelle le nom d'utilisateur présenté doit correspondre.
- **SHORTUSR** est une zone de l'enregistrement LDAP qui ne dépasse pas 12 caractères. La valeur de cette zone correspond à l'identité déclarée si l'authentification aboutit.
- **BASEDNU** représente le nom distinctif de base qui doit être utilisé pour les recherches dans LDAP.
- **BASEDNG** représente le nom distinctif de base des groupes dans LDAP.
- **AUTHORMD** définit le mécanisme utilisé pour résoudre l'appartenance à un groupe de l'utilisateur. Quatre options sont disponibles :
  - OS : recherchez les groupes associés au nom abrégé sur le système d'exploitation.
  - SEARCHGRP : recherchez l'utilisateur authentifié dans les entrées de groupe de LDAP.
  - SEARCHUSR : recherchez les informations sur l'appartenance à un groupe dans l'enregistrement de l'utilisateur authentifié.
  - SRCHGRPSN : recherchez le nom abrégé de l'utilisateur authentifié dans les entrées de groupe de LDAP (défini par la zone SHORTUSR).
- **GRPFIELD** représente l'attribut de l'enregistrement de groupe LDAP qui correspond à un nom simple. S'il est spécifié, il peut être utilisé pour définir des enregistrements d'autorisation.
- **CLASSUSR** représente la classe d'objet LDAP qui correspond à un utilisateur.
- **CLASSGRP** représente la classe d'objet LDAP qui correspond à un groupe.
- **FINDGRP** représente l'attribut de l'enregistrement LDAP qui correspond à l'appartenance à un groupe.

La nouvelle entrée peut être placée n'importe où dans le fichier, mais il peut s'avérer utile de placer les nouvelles entrées au début du fichier :



2. Associez la configuration LDAP à la définition de gestionnaire de files d'attente.

   Vous devez associer la configuration LDAP à la définition de gestionnaire de files d'attente. Immédiatement sous l'entrée `DEFINE AUTHINFO` figure une entrée `ALTER QMGR`. Modifiez l'entrée CONNAUTH pour qu'elle corresponde au nom AUTHINFO nouvellement créé. Par exemple, dans l'exemple précédent, `AUTHINFO(USE.LDAP)` étant défini, le nom est `USE.LDAP`. Vous devez donc remplacer `CONNAUTH('SYSTEM.DEFAULT.AUTHINFO.IDPWOS')` par `CONNAUTH('USE.LDAP')` :

Pour que le basculement vers LDAP soit immédiat, appelez une commande REFRESH SECURITY en ajoutant une ligne immédiatement après la commande ALTER QMGR :

*backup.mqsc

```
************************************************************************
* Script generated on 2020-10-21   at 11.48.32
* Script generated by user ' CallumJackso' on host 'LAPTOP-VLQKJ5UH'
* Queue manager name: qm1
* Queue manager platform: Windows
* Queue manager command level: (920/920)
* Command issued: dmpmqcfg -m qm1
************************************************************************
DEFINE AUTHINFO(USE.LDAP) +
   AUTHTYPE(IDPWLDAP) +
   CONNAME('ldap-service.ldap(389)') +
   LDAPUSER('cn=admin,dc=ibm,dc=com') +
   LDAPPWD('admin') +
   SECCOMM(NO) +
   USRFIELD('uid') +
   SHORTUSR('uid') +
   BASEDNU('ou=people,dc=ibm,dc=com') +
   AUTHORMD(SEARCHGRP) +
   BASEDNG('ou=groups,dc=ibm,dc=com') +
   GRPFIELD('cn') +
   CLASSGRP('groupOfUniqueNames') +
   FINDGRP('uniqueMember') +
   REPLACE
ALTER QMGR +
*  ALTDATE(2020-10-26) +
*  ALTTIME(11.43.11) +
   CCSID(850) +
   CERTLABL('default') +
   CLWLUSEQ(LOCAL) +
*  COMMANDQ(SYSTEM.ADMIN.COMMAND.QUEUE) +
   CONNAUTH('USE.LDAP') +
*  CRDATE(2020-10-26) +
*  CRTIME(11.43.11) +
*  QMID(qm1_2020-10-26_11.43.11) +
   SSLCRYP(' ') +
   SSLKEYR('/run/runmqserver/tls/key') +
   SUITEB(NONE) +
*  VERSION(09020000) +
   FORCE
REFRESH SECURITY
```

Que faire ensuite

Vous êtes maintenant prêt à mettre à jour la sauvegarde IBM MQ pour les informations d'autorisation LDAP.

LDAP - Partie 2 : mise à jour de la sauvegarde IBM MQ pour les informations d'autorisation LDAP

IBM MQ fournit des règles d'autorisation précises qui contrôlent l'accès aux objets IBM MQ. Si vous avez choisi LDAP pour l'authentification et l'autorisation, les règles d'autorisation peuvent être non valides et nécessiter une mise à jour.

Avant de commencer

Cette tâche suppose que vous avez mis à jour la sauvegarde du serveur LDAP.

Pourquoi et quand exécuter cette tâche

Cette tâche est-elle obligatoire ?

Si vous utilisez déjà LDAP pour l'authentification et l'autorisation, aucune modification n'est requise. Si vous n'êtes pas certain que LDAP est utilisé, reportez-vous à la rubrique «Facultatif : configuration de LDAP», à la page 42.

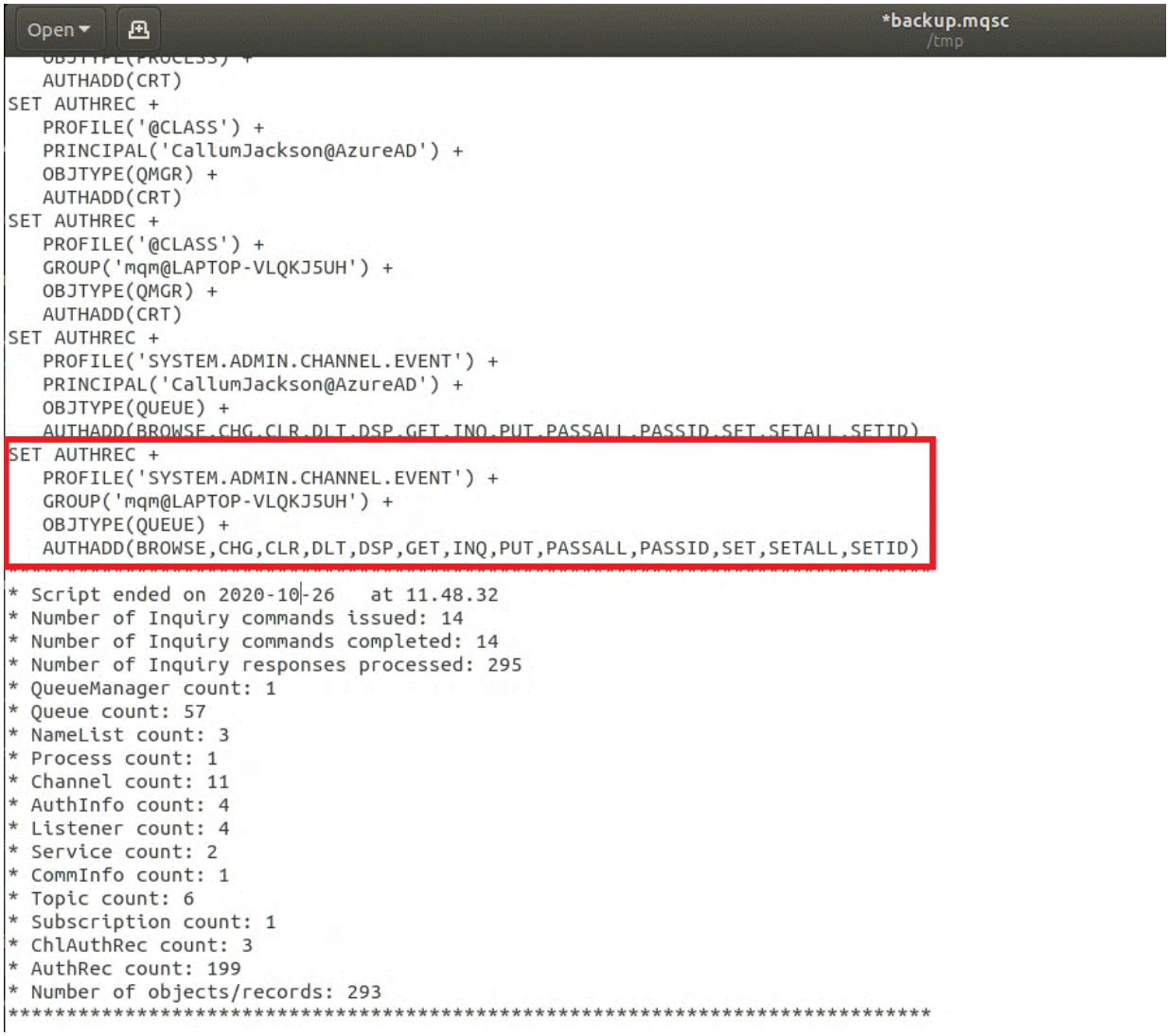
La mise à jour des informations d'autorisation LDAP se déroule en deux temps :

1. Suppression de toutes les autorisations existantes du fichier.
2. Définition des nouvelles informations d'autorisation pour LDAP.

Procédure

1. Supprimez toutes les autorisations existantes du fichier.

   Dans le fichier de sauvegarde, vers la fin du fichier, vous devez voir plusieurs entrées commençant par `SET AUTHREC` :



Recherchez les entrées existantes et supprimez-les. L'approche la plus directe consiste à supprimer toutes les règles SET AUTHREC existantes, puis à créer des entrées en fonction des entrées LDAP.

2. Définissez les nouvelles informations d'autorisation pour LDAP.

   En fonction de la configuration de votre gestionnaire de files d'attente et du nombre de ressources et de groupes, cette activité peut s'avérer fastidieuse ou simple. L'exemple suivant suppose que votre gestionnaire de files d'attente ne possède qu'une seule file d'attente appelée Q1 et que vous souhaitez que le groupe LDAP apps y ait accès.

```
SET AUTHREC GROUP('apps') OBJTYPE(QMGR) AUTHADD(ALL)
SET AUTHREC PROFILE('Q1') GROUP('apps') OBJTYPE(QUEUE) AUTHADD(ALL)
```

   La première commande AUTHREC ajoute le droit d'accès au gestionnaire de files d'attente et la deuxième permet d'accéder à la file d'attente. Si l'accès à une deuxième file d'attente est requis, une troisième commande AUTHREC est nécessaire, à moins que vous n'ayez décidé d'utiliser des caractères génériques pour fournir un accès plus générique.

   Voici un autre exemple. Si un groupe d'administrateurs (appelé admins) a besoin d'un accès complet au gestionnaire de files d'attente, ajoutez les commandes suivantes :

```
SET AUTHREC PROFILE('*') OBJTYPE(QUEUE) GROUP('admins') AUTHADD(ALL)
SET AUTHREC PROFILE('*') OBJTYPE(TOPIC) GROUP('admins') AUTHADD(ALL)
SET AUTHREC PROFILE('*') OBJTYPE(CHANNEL) GROUP('admins') AUTHADD(ALL)
SET AUTHREC PROFILE('*') OBJTYPE(CLNTCONN) GROUP('admins') AUTHADD(ALL)
```

```
SET AUTHREC PROFILE('*') OBJTYPE(AUTHINFO) GROUP('admins') AUTHADD(ALL)
SET AUTHREC PROFILE('*') OBJTYPE(LISTENER) GROUP('admins') AUTHADD(ALL)
SET AUTHREC PROFILE('*') OBJTYPE(NAMELIST) GROUP('admins') AUTHADD(ALL)
SET AUTHREC PROFILE('*') OBJTYPE(PROCESS) GROUP('admins') AUTHADD(ALL)
SET AUTHREC PROFILE('*') OBJTYPE(SERVICE) GROUP('admins') AUTHADD(ALL)
SET AUTHREC PROFILE('*') OBJTYPE(QMGR) GROUP('admins') AUTHADD(ALL)
```

Que faire ensuite

Vous êtes maintenant prêt à modifier les adresses IP et noms d'hôte dans la configuration IBM MQ.

OpenShift CD V 9.2.1 EUS Facultatif : modification des adresses IP et noms d'hôte dans la configuration IBM MQ

Des adresses IP et noms d'hôte ont peut-être été spécifiés pour la configuration IBM MQ. Dans certains cas, ils peuvent être conservés, alors que dans d'autres, ils doivent être mis à jour.

Avant de commencer

Cette tâche suppose que vous avez configuré LDAP.

Pourquoi et quand exécuter cette tâche

Cette tâche est-elle obligatoire ?

Déterminez au préalable si des adresses IP ou des noms d'hôte ont été spécifiés, en dehors de la configuration LDAP définies dans la section précédente. Pour cela, exécutez la commande suivante :

```
grep 'CONNAME\|LOCLADDR\|IPADDRV' -B 3  backup.mqsc
```

Exemple de sortie :

```
************************************************
DEFINE AUTHINFO(USE.LDAP) +
   AUTHTYPE(IDPWLDAP) +
   CONNAME('ldap-service.ldap(389)') +
--
DEFINE AUTHINFO('SYSTEM.DEFAULT.AUTHINFO.IDPWLDAP') +
   AUTHTYPE(IDPWLDAP) +
   ADOPTCTX(YES) +
   CONNAME(' ') +
--
   REPLACE
DEFINE AUTHINFO('SYSTEM.DEFAULT.AUTHINFO.CRLLDAP') +
   AUTHTYPE(CRLLDAP) +
   CONNAME(' ') +
```

Dans cet exemple, la recherche renvoie trois résultats. Un résultat correspond à la configuration LDAP définie précédemment. Ce résultat peut être ignoré car le nom d'hôte du serveur LDAP reste le même. Les deux autres résultats correspondent à des entrées de connexion vides et peuvent donc être également ignorés. Si vous ne disposez d'aucune autre entrée, vous pouvez ignorer le reste de cette rubrique.

Procédure

1. Vous devez comprendre les entrées renvoyées.

   IBM MQ peut inclure des adresses IP, des noms d'hôte et des ports dans de nombreux aspects de la configuration. Nous pouvons les classer en deux catégories :

   a. **Emplacement de ce gestionnaire de files d'attente** : informations d'emplacement que ce gestionnaire de files d'attente utilise ou publie, que d'autres gestionnaires de files d'attente ou applications au sein d'un réseau IBM MQ peuvent utiliser pour la connectivité.

b. **Emplacement des dépendances du gestionnaire de files d'attente** : emplacements des autres gestionnaires de files d'attente ou systèmes dont ce gestionnaire de files d'attente doit être conscient.

Ce scénario ne s'intéressant qu'aux modifications apportées à cette configuration de gestionnaire de files d'attente, nous ne traitons que les mises à jour de configuration de la catégorie (a). Toutefois, si l'emplacement de ce gestionnaire de files d'attente est référencé par d'autres gestionnaires de files d'attente ou applications, leur configuration peut avoir besoin d'être mise à jour conformément au nouvel emplacement de ce gestionnaire de files d'attente.

Deux objets clés peuvent contenir des informations à mettre à jour :

- Programmes d'écoute : ils représentent l'adresse réseau sur laquelle IBM MQ écoute.
- Canal CLUSTER RECEIVER : si le gestionnaire de files d'attente fait partie d'un cluster IBM MQ, cet objet existe. Il spécifie l'adresse réseau à laquelle les autres gestionnaires de files d'attente peuvent se connecter.

2. Dans la sortie d'origine de la commande `grep 'CONNAME\|LOCLADDR\|IPADDRV' -B 3 backup.mqsc` , identifiez si des canaux CLUSTER RECEIVER sont définis. Si tel est le cas, mettez à jour les adresses IP.

   Pour déterminer si des canaux CLUSTER RECEIVER sont définis, recherchez les entrées contenant CHLTYPE(CLUSRCVR) dans la sortie d'origine :

   ```
   DEFINE CHANNEL(ANY_NAME) +
      CHLTYPE(CLUSRCVR) +
   ```

   Si des entrées existent, mettez à jour CONNAME avec la route IBM MQ Red Hat OpenShift. Cette valeur est basée sur l'environnement Red Hat OpenShift et utilise une syntaxe prévisible :

   ```
   queue_manager_resource_name-ibm-mq-qm-openshift_project_name.openshift_app_route_hostname
   ```

   Par exemple, si le déploiement du gestionnaire de files d'attente est nommé qm1 dans l'espace de nom cp4i et que *openshift_app_route_hostname* est `apps.callumj.icp4i.com`, l'URL de la route est la suivante :

   ```
   qm1-ibm-mq-qm-cp4i.apps.callumj.icp4i.com
   ```

   Le numéro de port du chemin est généralement 443. À moins que votre administrateur Red Hat OpenShift ne vous indique différemment, il s'agit normalement de la valeur correcte. A l'aide de ces informations, mettez à jour les zones CONNAME. Exemple :

   ```
   CONNAME('qm1-ibm-mq-qm-cp4i.apps.callumj.icp4i.com(443)')
   ```

   Dans la sortie d'origine de la commande `grep 'CONNAME\|LOCLADDR\|IPADDRV' -B 3 backup.mqsc` , vérifiez s'il existe des entrées pour LOCLADDR ou IPADDRV. Si c'est le cas, supprimez-les. Elles ne sont pas pertinentes dans un environnement de conteneur.

Que faire ensuite

Vous êtes maintenant prêt à mettre à jour la configuration du gestionnaire de files d'attente pour un environnement de conteneur.

OpenShift CD V 9.2.1 EUS Mise à jour de la configuration du gestionnaire de files d'attente pour un environnement de conteneur

Lors de l'exécution dans un conteneur, certains aspects de la configuration sont définis par le conteneur et peuvent être en conflit avec la configuration exportée.

Avant de commencer

Cette tâche suppose que vous avez modifié la configuration des adresses IP et noms d'hôte d'IBM MQ.

Pourquoi et quand exécuter cette tâche

Les aspects suivants de la configuration sont définis par le conteneur :

- Définitions du programme d'écoute (qui correspondent aux ports exposés).
- Emplacement de tout magasin TLS potentiel.

Par conséquent, vous devez mettre à jour la configuration exportée :

1. Supprimez les définitions du programme d'écoute.
2. Définissez l'emplacement du référentiel de clés TLS.

Procédure

1. Supprimez les définitions du programme d'écoute.

   Dans la configuration de sauvegarde, recherchez `DEFINE LISTENER`. Cette section doit se trouver entre les définitions `AUTHINFO` et `SERVICE`. Mettez en évidence la zone, puis supprimez-la.

*backup.mqsc

```
*  ALTDATE(2020-11-26) +
*  ALTTIME(11.43.28) +
   REPLACE
DEFINE AUTHINFO('SYSTEM.DEFAULT.AUTHINFO.CRLLDAP') +
   AUTHTYPE(CRLLDAP) +
   CONNAME(' ') +
*  ALTDATE(2020-10-26) +
*  ALTTIME(11.43.28) +
   REPLACE
DEFINE LISTENER('SYSTEM.DEFAULT.LISTENER.LU62') +
   TRPTYPE(LU62) +
   CONTROL(MANUAL) +
*  ALTDATE(2020-10-26) +
*  ALTTIME(11.43.28) +
   REPLACE
DEFINE LISTENER('SYSTEM.DEFAULT.LISTENER.NETBIOS') +
   TRPTYPE(NETBIOS) +
   CONTROL(MANUAL) +
   LOCLNAME(' ') +
*  ALTDATE(2020-10-26) +
*  ALTTIME(11.43.28) +
   REPLACE
DEFINE LISTENER('SYSTEM.DEFAULT.LISTENER.SPX') +
   TRPTYPE(SPX) +
   CONTROL(MANUAL) +
*  ALTDATE(2020-10-26) +
*  ALTTIME(11.43.28) +
   REPLACE
DEFINE LISTENER('SYSTEM.DEFAULT.LISTENER.TCP') +
   TRPTYPE(TCP) +
   CONTROL(MANUAL) +
*  ALTDATE(2020-10-26) +
*  ALTTIME(11.43.28) +
   REPLACE
DEFINE SERVICE('SYSTEM.AMQP.SERVICE') +
   CONTROL(QMGR) +
   SERVTYPE(SERVER) +
   STARTCMD('+MQ_INSTALL_PATH+\bin\amqp.bat') +
   STARTARG('start -m +QMNAME+ -d "+MQ_Q_MGR_DATA_PATH+\."
   STOPCMD('+MQ_INSTALL_PATH+\bin64\endmqsde.exe') +
```

2. Définissez l'emplacement du référentiel de clés TLS.

   La sauvegarde du gestionnaire de files d'attente contient la configuration TLS de l'environnement d'origine. Cela est différent de l'environnement de conteneur, et quelques mises à jour sont donc nécessaires :

   - Remplacez l'entrée **CERTLABL** par `default`
   - Remplacez l'emplacement du référentiel de clés TLS (**SSLKEYR**) par `/run/runmqserver/tls/key`

Pour rechercher l'emplacement de l'attribut **SSLKEYR** dans le fichier, recherchez **SSLKEYR**. En général, il n'existe qu'une seule entrée. Si plusieurs entrées sont détectées, veillez à bien éditer l'objet **QMGR** comme indiqué dans l'illustration suivante :

*backup.mqsc

```
************************************************************************
* Script generated on 2020-10-21   at 11.48.32
* Script generated by user ' CallumJackso' on host 'LAPTOP-VLQKJ5UH'
* Queue manager name: qm1
* Queue manager platform: Windows
* Queue manager command level: (920/920)
* Command issued: dmpmqcfg -m qm1
************************************************************************
DEFINE AUTHINFO(USE.LDAP) +
   AUTHTYPE(IDPWLDAP) +
   CONNAME('ldap-service.ldap(389)') +
   LDAPUSER('cn=admin,dc=ibm,dc=com') +
   LDAPPWD('admin') +
   SECCOMM(NO) +
   USRFIELD('uid') +
   SHORTUSR('uid') +
   BASEDNU('ou=people,dc=ibm,dc=com') +
   AUTHORMD(SEARCHGRP) +
   BASEDNG('ou=groups,dc=ibm,dc=com') +
   GRPFIELD('cn') +
   CLASSGRP('groupOfUniqueNames') +
   FINDGRP('uniqueMember') +
   REPLACE
ALTER QMGR +
*  ALTDATE(2020-10-26) +
*  ALTTIME(11.43.11) +
   CCSID(850) +
   CERTLABL('default') +
   CLWLUSEQ(LOCAL) +
*  COMMANDQ(SYSTEM.ADMIN.COMMAND.QUEUE) +
   CONNAUTH('USE.LDAP') +
*  CRDATE(2020-10-26) +
*  CRTIME(11.43.11) +
*  QMID(qm1_2020-10-26_11.43.11) +
   SSLCRYP(' ') +
   SSLKEYR('/run/runmqserver/tls/key') +
   SUITEB(NONE) +
*  VERSION(09020000) +
   FORCE
REFRESH SECURITY
```

Que faire ensuite

Vous êtes maintenant prêt à sélectionner l'architecture cible d'IBM MQ exécutée en conteneurs.

OpenShift CD V 9.2.1 EUS Sélection de l'architecture haute disponibilité cible pour IBM MQ exécuté en conteneurs

Choisissez entre une instance unique (un seul pod Kubernetes) et plusieurs instances (deux pods) pour répondre à vos besoins en matière de haute disponibilité.

Avant de commencer

Cette tâche suppose que vous avez mis à jour la configuration du gestionnaire de files d'attente pour un environnement de conteneur.

Pourquoi et quand exécuter cette tâche

Le IBM MQ Operator offre deux options de haute disponibilité :

- **Instance unique** : Un conteneur unique (Pod) est démarré et il est de la responsabilité de Red Hat OpenShift de redémarrer en cas d'échec. En raison des caractéristiques d'un ensemble avec état dans Kubernetes, il arrive parfois que cette reprise en ligne soit longue ou qu'elle requière une action administrative.
- **Multi-instance** : deux conteneurs (chacun dans un pod distinct) sont démarrés ; l'un en mode actif et l'autre en mode de secours. Cette topologie permet une reprise en ligne bien plus rapide. Elle requiert un système de fichiers RWM (Read Write Many) qui répond aux exigences d'IBM MQ.

Dans cette tâche, vous vous contentez de choisir l'architecture haute disponibilité cible. Les étapes de configuration de l'architecture choisie sont décrites dans une tâche ultérieure de ce scénario («Création du gestionnaire de files d'attente dans Red Hat OpenShift», à la page 56).

Procédure

1. Examinez les deux options.

   Pour une description complète de ces deux options, voir «Haute disponibilité pour IBM MQ dans les conteneurs», à la page 16.
2. Sélectionnez l'architecture haute disponibilité cible.

   Si vous ne savez pas quelle option choisir, commencez par l'option **Instance unique** et vérifiez si elle répond à vos besoins en matière de haute disponibilité.

Que faire ensuite

Vous êtes maintenant prêt à créer les ressources du gestionnaire de files d'attente.

OpenShift CD V 9.2.1 EUS Création des ressources du gestionnaire de files d'attente

Importez la configuration IBM MQ, ainsi que les certificats et les clés TLS, dans l'environnement Red Hat OpenShift.

Avant de commencer

Cette tâche suppose que vous avez sélectionné l'architecture cible pour IBM MQ exécuté en conteneurs.

Pourquoi et quand exécuter cette tâche

Dans les sections précédentes, vous avez extrait, mis à jour et défini deux ressources :

- Configuration de IBM MQ
- Certificats et clés TLS

Vous devez importer ces ressources dans l'environnement Red Hat OpenShift avant le déploiement du gestionnaire de files d'attente.

Procédure

1. Importez la configuration IBM MQ dans Red Hat OpenShift.

   Les instructions ci-après supposent que la configuration IBM MQ se trouve dans le répertoire de travail, dans un fichier appelé `backup.mqsc`. Sinon, vous devez personnaliser le nom de ce fichier en fonction de votre environnement.

   a) Connectez-vous à votre cluster à l'aide de la commande `oc login`.

   b) Chargez la configuration IBM MQ dans une `configmap`.

      Exécutez ensuite la commande suivante :

      ```
      oc create configmap my-mqsc-migrated --from-file=backup.mqsc
      ```

   c) Vérifiez que le fichier a bien été chargé.

      Exécutez ensuite la commande suivante :

      ```
      oc describe configmap my-mqsc-migrated
      ```

2. Importez les ressources TLS d'IBM MQ

   Comme indiqué dans la rubrique «Facultatif : extraction et acquisition des clés et certificats du gestionnaire de files d'attente», à la page 40, TLS peut être requis pour le déploiement du gestionnaire de files d'attente. Si tel est le cas, vous devez déjà avoir un certain nombre de fichiers se terminant par `.crt` et `.key`. Vous devez les ajouter dans les secrets Kubernetes pour que le gestionnaire de files d'attente y fasse référence lors de la phase de déploiement.

   Par exemple, si vous disposez d'une clé et d'un certificat pour le gestionnaire de files d'attente, ils peuvent s'appeler :

   - `qmgr.crt`
   - `qmgr.key`

   Pour importer ces fichiers, exécutez la commande suivante :

   ```
   oc create secret tls my-tls-migration --cert=qmgr.crt --key=qmgr.key
   ```

   Kubernetes fournit cet utilitaire utile lorsque vous importez une clé publique et privée correspondante. Pour ajouter d'autres certificats, par exemple, dans le magasin de clés de confiance du gestionnaire de files d'attente, exécutez la commande suivante :

   ```
   oc create secret generic my-extra-tls-migration --from-file=comma_separated_list_of_files
   ```

   Par exemple, si les fichiers à importer sont `trust1.crt`, `trust2.crt` et `trust3.crt`, la commande est la suivante :

   ```
   oc create secret generic my-extra-tls-migration --from-file=trust1.crt,trust2.crt,trust3.crt
   ```

Que faire ensuite

Vous êtes maintenant prêt à créer le gestionnaire de files d'attente sur Red Hat OpenShift.

Création du gestionnaire de files d'attente dans Red Hat OpenShift

Déployez un gestionnaire de files d'attente à instance unique ou multi-instance dans Red Hat OpenShift.

Avant de commencer

Cette tâche suppose que vous avez créé les ressources du gestionnaire de files d'attente et installé le IBM MQ Operator dans Red Hat OpenShift.

Pourquoi et quand exécuter cette tâche

Comme indiqué dans la rubrique «Sélection de l'architecture haute disponibilité cible pour IBM MQ exécuté en conteneurs», à la page 55, deux topologies de déploiement sont possibles. Par conséquent, cette rubrique fournit deux modèles différents :

- Déployez un gestionnaire de files d'attente à instance unique.
- Déployez un gestionnaire de files d'attente multi-instance.

Important : N'exécutez que l'un des deux modèles, en fonction de la topologie de votre choix.

Procédure

- Déployez un gestionnaire de files d'attente à instance unique.

  Le gestionnaire de files d'attente migré est déployé sur Red Hat OpenShift à l'aide d'un fichier YAML. En voici un exemple, basé sur les noms utilisés dans les rubriques précédentes :

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: qm1
spec:
  version: 9.2.5.0-r3
  license:
    accept: true
    license: L-RJON-C7QG3S
    use: "Production"
  pki:
    keys:
    - name: default
      secret:
        secretName: my-tls-migration
        items:
          - tls.key
          - tls.crt
  web:
    enabled: true
  queueManager:
    name: QM1
    mqsc:
      - configMap:
          name: my-mqsc-migrated
          items:
            - backup.mqsc
```

  Selon les étapes que vous avez effectuées, il peut être nécessaire de personnaliser le fichier YAML précédent. Pour vous y aider, vous trouverez ci-dessous une explication de ce fichier YAML :

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: qm1
```

  Définit l'objet Kubernetes, son type et son nom. La seule zone nécessitant une personnalisation est la zone `name`.

```
 spec:
  version: 9.2.5.0-r3
  license:
    accept: true
    license: L-RJON-C7QG3S
    use: "Production"
```

Correspond aux informations de version et de licence du déploiement. Pour les personnaliser, utilisez les informations fournies dans la rubrique «Référence relative à l'octroi de licence pour mq.ibm.com/v1beta1», à la page 131.

```
  pki:
    keys:
    - name: default
      secret:
        secretName: my-tls-migration
        items:
          - tls.key
          - tls.crt
```

Pour que le gestionnaire de files d'attente soit configuré afin d'utiliser TLS, il doit faire référence aux certificats et aux clés appropriés. La zone `secretName` fait référence au secret Kubernetes créé dans la section Importation des ressources IBM MQ du TLS, et la liste des éléments (`tls.key` et `tls.crt`) est le nom standard que Kubernetes attribue lors de l'utilisation de la syntaxe `oc create secret tls`. Si vous devez ajouter des certificats supplémentaires dans le magasin de clés de confiance, vous pouvez procéder de la même manière, mais les éléments sont les noms de fichier correspondants utilisés lors de l'importation. Par exemple, le code suivant peut être utilisé pour créer les certificats du magasin de clés de confiance :

```
oc create secret generic my-extra-tls-migration --from-file=trust1.crt,trust2.crt,trust3.crt
```

```
  pki:
    trust:
    - name: default
      secret:
        secretName: my-extra-tls-migration
        items:
          - trust1.crt
          - trust2.crt
          - trust3.crt
```

Important : Si TLS n'est pas requis, supprimez la section TLS du fichier YAML.

```
  web:
    enabled: true
```

Active la console Web pour le déploiement

```
  queueManager:
    name: QM1
```

Spécifie QM1 comme nom de gestionnaire de files d'attente. Le gestionnaire de files d'attente est personnalisé en fonction de vos exigences (par exemple, le nom d'origine du gestionnaire de files d'attente).

```
    mqsc:
      - configMap:
          name: my-mqsc-migrated
          items:
            - backup.mqsc
```

Le code précédent extrait la configuration de gestionnaire de files d'attente importée dans la section Importez la configuration IBM MQ. Si vous avez utilisé des noms différents, vous devez modifier `my-mqsc-migrated` et `backup.mqsc`.

Notez que l'exemple YAML suppose que la classe de stockage par défaut pour l'environnement Red Hat OpenShift soit définie comme une classe de stockage RWX ou RWO. Si aucune valeur par défaut n'est définie dans votre environnement, vous devez spécifier la classe de stockage à utiliser. Pour cela, vous pouvez étendre le fichier YAML comme suit :

```
  queueManager:
    name: QM1
    storage:
```

```
      defaultClass: my_storage_class
      queueManager:
        type: persistent-claim
```

Ajoutez le texte mis en évidence, avec l'attribut de classe personnalisé en fonction de votre environnement. Pour découvrir les noms de classe de stockage dans votre environnement, exécutez la commande suivante :

```
oc get storageclass
```

Voici un exemple de sortie renvoyée par cette commande :

```
NAME          PROVISIONER             RECLAIMPOLICY
aws-efs       openshift.org/aws-efs   Delete
gp2 (default) kubernetes.io/aws-ebs   Delete
```

Le code suivant montre comment référencer la configuration IBM MQ importée dans la section Importez la configuration IBM MQ. Si vous avez utilisé des noms différents, vous devez modifier `my-mqsc-migrated` et `backup.mqsc`.

```
    mqsc:
      - configMap:
          name: my-mqsc-migrated
          items:
            - backup.mqsc
```

Vous avez déployé votre gestionnaire de files d'attente à instance unique. Le modèle est terminé. Vous êtes maintenant prêt à vérifier le nouveau déploiement de conteneur.

- Déployez un gestionnaire de files d'attente multi-instance.

  Le gestionnaire de files d'attente migré est déployé sur Red Hat OpenShift à l'aide d'un fichier YAML. L'exemple suivant reprend les noms utilisés dans les sections précédentes.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: qm1mi
spec:
  version: 9.2.5.0-r3
  license:
    accept: true
    license: L-RJON-C7QG3S
    use: "Production"
  pki:
    keys:
    - name: default
      secret:
        secretName: my-tls-migration
        items:
          - tls.key
          - tls.crt
  web:
    enabled: true
  queueManager:
    name: QM1
    availability: MultiInstance
    storage:
      defaultClass: aws-efs
      persistedData:
        enabled: true
      queueManager:
        enabled: true
      recoveryLogs:
        enabled: true
    mqsc:
      - configMap:
          name: my-mqsc-migrated
          items:
            - backup.mqsc
```

Voici une explication de ce fichier YAML. La majorité de la configuration suit la même approche que le déploiement d'un gestionnaire de files d'attente à instance unique. Par conséquent, seuls les aspects de disponibilité et de stockage du gestionnaire de files d'attente sont expliqués ici.

```
queueManager:
    name: QM1
    availability: MultiInstance
```

Ceci indique le nom du gestionnaire de files d'attente sous la forme `QM1` et définit le déploiement comme étant `MultiInstance` au lieu de l'instance unique par défaut.

```
    storage:
      defaultClass: aws-efs
      persistedData:
        enabled: true
      queueManager:
        enabled: true
      recoveryLogs:
        enabled: true
```

Un gestionnaire de files d'attente multi-instance IBM MQ dépend du stockage RWX. Par défaut, un gestionnaire de files d'attente est déployé en mode instance unique et des options de stockage supplémentaires sont donc requises lors d'un passage au mode multi-instance. Dans l'exemple de fichier YAML précédent, trois volumes persistants de stockage et une classe de volume persistant sont définis. Cette classe de volume persistant doit être une classe de stockage RWX. Si vous n'êtes pas certain des noms de classe de stockage dans votre environnement, vous pouvez exécuter la commande suivante pour les découvrir :

```
oc get storageclass
```

Voici un exemple de sortie renvoyée par cette commande :

```
NAME          PROVISIONER             RECLAIMPOLICY
aws-efs       openshift.org/aws-efs   Delete
gp2 (default) kubernetes.io/aws-ebs   Delete
```

Le code suivant montre comment référencer la configuration IBM MQ importée dans la section Importez la configuration IBM MQ. Si vous avez utilisé des noms différents, vous devez modifier `my-mqsc-migrated` et `backup.mqsc`.

```
    mqsc:
      - configMap:
          name: my-mqsc-migrated
          items:
            - backup.mqsc
```

Vous avez déployé votre gestionnaire de files d'attente multi-instance. Le modèle est terminé. Vous êtes maintenant prêt à vérifier le nouveau déploiement de conteneur.

Vérification du nouveau déploiement de conteneur

Maintenant que IBM MQ est déployé sur Red Hat OpenShift, vous pouvez vérifier l'environnement à l'aide des exemples IBM MQ.

Avant de commencer

Cette tâche suppose que vous avez créé le gestionnaire de files d'attente sous Red Hat OpenShift.

Important : Cette tâche suppose que TLS n'est pas activé dans le gestionnaire de files d'attente.

Pourquoi et quand exécuter cette tâche

Dans cette tâche, vous exécutez les exemples IBM MQ depuis le conteneur du gestionnaire de files d'attente migré. Toutefois, vous pouvez choisir d'utiliser vos propres applications exécutées depuis un autre environnement.

Vous devez disposer des informations suivantes :

- Nom d'utilisateur LDAP
- Mot de passe LDAP
- Nom du canal IBM MQ
- Nom de la file d'attente

Cet exemple de code utilise les paramètres ci-après. Notez que vos paramètres seront différents.

- Nom d'utilisateur LDAP : mqapp
- Mot de passe LDAP : mqapp
- Nom du canal IBM MQ : DEV.APP.SVRCONN
- Nom de la file d'attente : Q1

Procédure

1. Exécutez la commande Exec dans le conteneur IBM MQ en cours d'exécution.

   Utilisez la commande suivante :

   ```
   oc exec -it qm1-ibm-mq-0 /bin/bash
   ```

   où `qm1-ibm-mq-0` représente le pod que nous avons déployé dans la rubrique «Création du gestionnaire de files d'attente dans Red Hat OpenShift», à la page 56. Si votre déploiement porte un autre nom, personnalisez cette valeur.
2. Envoyez un message.

   Exécutez les commandes suivantes :

   ```
   cd /opt/mqm/samp/bin
   export IBM MQSAMP_USER_ID=mqapp
   export IBM MQSERVER=DEV.APP.SVRCONN/TCP/'localhost(1414)'
   ./amqsputc Q1 QM1
   ```

   Vous êtes invité à entrer un mot de passe, avant de pouvoir envoyer un message.
3. Vérifiez que le message a bien été reçu.

   Exécutez l'exemple GET :

   ```
   ./amqsgetc Q1 QM1
   ```

Résultats

Vous avez terminé le «Migration d'IBM MQ vers IBM Cloud Pak for Integration», à la page 37.

Que faire ensuite

Utilisez les informations suivantes pour vous aider dans des scénarios de migration plus complexes :

Migration des messages en file d'attente
Pour migrer des messages en file d'attente existants, suivez les conseils de la rubrique suivante pour l'exportation et l'importation de messages une fois que le nouveau gestionnaire de files d'attente est en place : Utilisation de l'utilitaire dmpmqmsg entre deux systèmes.

Connexion à IBM MQ à partir de l'environnement Red Hat OpenShift

Le gestionnaire de files d'attente déployé peut être exposé aux clients IBM MQ et aux gestionnaires de files d'attente en dehors de l'environnement Red Hat OpenShift. Le processus dépend de la version de IBM MQ qui se connecte à l'environnement Red Hat OpenShift. Voir «Configuration d'une route pour la connexion à un gestionnaire de files d'attente depuis l'extérieur d'un cluster Red Hat OpenShift», à la page 113.

OpenShift CP4I Installation et désinstallation de IBM MQ Operator sous Red Hat OpenShift

IBM MQ Operator peut être installé sur Red Hat OpenShift à l'aide de l'opérateur Hub.

Procédure

- «Dépendances pour IBM MQ Operator», à la page 10.
- «Droits d'accès au cluster requis par IBM MQ Operator», à la page 11.
- «Installation de IBM MQ Operator à l'aide de la console Web Red Hat OpenShift», à la page 62.
- «Installation de IBM MQ Operator à l'aide de l'interface de ligne de commande Red Hat OpenShift», à la page 64.
- «Installation d'IBM MQ Operator dans un environnement isolé physiquement», à la page 68.

Tâches associées

«Désinstallation de IBM MQ Operator à l'aide de la console Web Red Hat OpenShift», à la page 64
Vous pouvez utiliser la console Web Red Hat OpenShift pour désinstaller IBM MQ Operator de Red Hat OpenShift.

«Désinstallation de IBM MQ Operator à l'aide de l'interface de ligne de commande Red Hat OpenShift», à la page 67
Vous pouvez utiliser l'interface de ligne de commande Red Hat OpenShift pour désinstaller IBM MQ Operator de Red Hat OpenShift. Il existe des différences dans le processus de désinstallation, selon que IBM MQ Operator est installé dans un seul espace de nom, ou installé et disponible pour tous les espaces de nom sur le cluster.

OpenShift CP4I Installation de IBM MQ Operator à l'aide de la console Web Red Hat OpenShift

IBM MQ Operator peut être installé sur Red Hat OpenShift à l'aide de l'opérateur Hub.

Avant de commencer

Connectez-vous à votre console Web de cluster Red Hat OpenShift.

Procédure

1. EUS

   Facultatif : Ajoutez les opérateurs de services communs IBM à la liste des opérateurs installables.

   Remarque :

   Cette étape s'applique aux éditions de IBM MQ Operator 1.5 et versions antérieures. L'étape ajoute un catalogue de services communs distinct. Pour les éditions ultérieures de l'opérateur, les services communs sont inclus dans le catalogue IBM.

   a) Cliquez sur l'icône plus en haut à droite de l'écran. La boîte de dialogue **Import YAML** s'ouvre.

   b) Collez la ressource de définition suivante dans la boîte de dialogue :

   ```
   apiVersion: operators.coreos.com/v1alpha1
   kind: CatalogSource
   metadata:
   ```

```
  name: opencloud-operators
  namespace: openshift-marketplace
spec:
  displayName: IBMCS Operators
  publisher: IBM
  sourceType: grpc
  image: icr.io/cpopen/ibm-common-service-catalog:latest
  updateStrategy:
    registryPoll:
      interval: 45m
```

c) Cliquez sur **Créer**.

2. Ajoutez les opérateurs IBM à la liste des opérateurs pouvant être installés.
   a) Cliquez sur l'icône plus en haut à droite de l'écran. La boîte de dialogue **Import YAML** s'ouvre.
   b) Collez la ressource de définition suivante dans la boîte de dialogue :

```
apiVersion: operators.coreos.com/v1alpha1
kind: CatalogSource
metadata:
  name: ibm-operator-catalog
  namespace: openshift-marketplace
spec:
  displayName: IBM Operator Catalog
  image: icr.io/cpopen/ibm-operator-catalog:latest
  publisher: IBM
  sourceType: grpc
  updateStrategy:
    registryPoll:
      interval: 45m
```

   c) Cliquez sur **Créer**.
3. Créez un espace de nom à utiliser pour IBM MQ Operator

   IBM MQ Operator peut être installé dans un espace de nom unique ou dans tous les espaces de nom. Cette étape n'est nécessaire que si vous voulez procéder à l'installation dans un espace de nom particulier qui n'existe pas encore.
   a) Depuis le panneau de navigation, cliquez sur **Home** > **Projects**.

      La page Projects s'affiche.
   b) Cliquez sur **Create Project**. Une zone Create Project s'affiche.
   c) Entrez les détails de l'espace de nom que vous créez. Par exemple, vous pouvez spécifier le nom "ibm-mq".
   d) Cliquez sur **Créer**. L'espace de nom de IBM MQ Operator est créé.
4. Installez IBM MQ Operator.
   a) Depuis le panneau de navigation, cliquez sur **Operators** > **OperatorHub**.

      La page OperatorHub s'affiche.
   b) Dans la zone **All Items**, entrez "IBM MQ".

      L'entrée de catalogue IBM MQ est affichée.
   c) Sélectionnez **IBM MQ**.

      La fenêtre IBM MQ s'ouvre.
   d) Cliquez sur **Install**.

      La page Create Operator Subscription s'affiche.
   e) Consultez la rubrique «Versions prises en charge pour IBM MQ Operator», à la page 7 pour déterminer le canal d'opérateur à sélectionner.
   f) Définissez le mode d'installation sur l'espace de nom spécifique que vous avez créé ou sur la portée à l'échelle du cluster.

      Il est recommandé de choisir la portée à l'échelle du cluster, car l'installation de différentes versions d'un opérateur dans différents espaces de nom peut entraîner des problèmes. Les opérateurs sont conçus pour être des extensions du plan de contrôle.
   g) Cliquez sur **Subscribe**.

IBM MQ apparaît dans la page Installed Operators.

h) Cliquez sur le statut de l'opérateur dans la page Installed Operators ; il devient Succeeded une fois l'installation terminée.

Que faire ensuite

«Préparation de votre projet Red Hat OpenShift pour IBM MQ à l'aide de la console Web Red Hat OpenShift», à la page 83

OpenShift CP4I *Désinstallation de IBM MQ Operator à l'aide de la console Web Red Hat OpenShift*

Vous pouvez utiliser la console Web Red Hat OpenShift pour désinstaller IBM MQ Operator de Red Hat OpenShift.

Avant de commencer

Connectez-vous à la console Web de votre cluster Red Hat OpenShift.

Si IBM MQ Operator est installé sur tous les projets/espaces de nom du cluster, répétez les étapes 1 à 5 de la procédure suivante pour chaque projet sur lequel vous souhaitez supprimer des gestionnaires de files d'attente.

Procédure

1. Sélectionnez **Operators** > **Installed Operators**.
2. Dans la liste déroulante **Project**, sélectionnez un projet.
3. Cliquez sur l'opérateur **IBM MQ**.
4. Cliquez sur l'onglet **Queue Managers** pour afficher les gestionnaires de files d'attente gérés par IBM MQ Operator.
5. Supprimez un ou plusieurs gestionnaires de files d'attente.

   Notez que, bien que ces gestionnaires de files d'attente continuent à s'exécuter, ils risquent de ne pas fonctionner comme prévu sans IBM MQ Operator.
6. Facultatif : Si nécessaire, répétez les étapes 1 à 5 pour chaque projet dans lequel vous souhaitez supprimer des gestionnaires de files d'attente.
7. Revenez à **Operators** > **Installed Operators**.
8. En regard de l'opérateur **IBM MQ**, cliquez sur le menu à trois points et sélectionnez **Uninstall Operator**.
9. Si vous utilisez Red Hat OpenShift Container Platform 4.7, vous devrez peut-être supprimer manuellement le point d'ancrage Web de validation à partir de la ligne de commande :

```
oc delete validatingwebhookconfiguration namespace.validator.queuemanagers.mq.ibm.com
```

OpenShift CP4I Installation de IBM MQ Operator à l'aide de l'interface de ligne de commande Red Hat OpenShift

IBM MQ Operator peut être installé sur Red Hat OpenShift à l'aide de l'opérateur Hub.

Avant de commencer

Connectez-vous à l'interface de ligne de commande Red Hat OpenShift à l'aide de **oc login**. Pour pouvoir effectuer ces étapes, vous devez être un administrateur de cluster.

Procédure

1. EUS

Facultatif : Créez un **CatalogSource** pour les opérateurs de services communs IBM.

Remarque :

Cette étape s'applique aux éditions de IBM MQ Operator 1.5 et versions antérieures. L'étape ajoute un catalogue de services communs distinct. Pour les éditions ultérieures de l'opérateur, les services communs sont inclus dans le catalogue IBM.

a) Créez un fichier YAML définissant la ressource **CatalogSource**.

Créez un fichier nommé "operator-source-cs.yaml" dont le contenu est le suivant :

```
apiVersion: operators.coreos.com/v1alpha1
kind: CatalogSource
metadata:
  name: opencloud-operators
  namespace: openshift-marketplace
spec:
  displayName: IBMCS Operators
  publisher: IBM
  sourceType: grpc
  image: icr.io/cpopen/ibm-common-service-catalog:latest
  updateStrategy:
    registryPoll:
      interval: 45m
```

b) Appliquez le **CatalogSource** au serveur.

```
oc apply -f operator-source-cs.yaml -n openshift-marketplace
```

2. Créer un **CatalogSource** pour les opérateurs IBM

a) Créez un fichier YAML définissant la ressource **CatalogSource**

Créez un fichier nommé "operator-source-ibm.yaml" dont le contenu est le suivant :

```
apiVersion: operators.coreos.com/v1alpha1
kind: CatalogSource
metadata:
  name: ibm-operator-catalog
  namespace: openshift-marketplace
spec:
  displayName: IBM Operator Catalog
  image: icr.io/cpopen/ibm-operator-catalog:latest
  publisher: IBM
  sourceType: grpc
  updateStrategy:
    registryPoll:
      interval: 45m
```

b) Appliquez le **CatalogSource** au serveur.

```
oc apply -f operator-source-ibm.yaml -n openshift-marketplace
```

3. Créez un espace de nom à utiliser pour IBM MQ Operator

IBM MQ Operator peut être installé dans un espace de nom unique ou dans tous les espaces de nom. Cette étape n'est nécessaire que si vous voulez procéder à l'installation dans un espace de nom particulier qui n'existe pas encore.

```
oc new-project ibm-mq
```

4. Affichez la liste des opérateurs disponibles dans le cluster depuis OperatorHub.

```
oc get packagemanifests -n openshift-marketplace
```

5. Inspectez IBM MQ Operator pour vérifier les modes d'installation qu'il prend en charge et les canaux disponibles

```
oc describe packagemanifests ibm-mq -n openshift-marketplace
```

6. Créer un fichier YAML d'objet **OperatorGroup**

Une **OperatorGroup** est une ressource OLM qui sélectionne les espaces de nom cible dans lesquels générer l'accès RBAC requis pour tous les opérateurs du même espace de nom que **OperatorGroup**.

L'espace de nom auquel vous souscrirez l'opérateur doit avoir un **OperatorGroup** qui correspond au **InstallMode**de l'opérateur, soit le mode `AllNamespaces` ou `SingleNamespace`. Si l'opérateur que vous avez l'intention d'installer utilise `AllNamespaces`, l'espace de nom `openshift-operators` dispose déjà d'un **OperatorGroup** approprié.

Toutefois, si l'opérateur utilise le mode `Espace de nom unique` et que vous ne disposez pas déjà d'un **OperatorGroup** approprié, vous devez en créer un.

a) Créez un fichier nommé "mq-operator-group.yaml" dont le contenu est le suivant :

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: <operatorgroup_name>
  namespace: <namespace_name>
spec:
  targetNamespaces:
  - <namespace_name>
```

b) Création de l'objet **OperatorGroup**

```
oc apply -f mq-operator-group.yaml
```

7. Créez un fichier YAML d'objet **Subscription** pour abonner un espace de nom à IBM MQ Operator

a) Consultez la rubrique «Versions prises en charge pour IBM MQ Operator», à la page 7 pour déterminer le canal d'opérateur à sélectionner.

b) Créez un fichier appelé "mq-sub. Yaml" avec le contenu suivant, mais modifiez **channel** pour qu'il corresponde au canal de la version de IBM MQ Operator que vous souhaitez installer.

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: ibm-mq
  namespace: openshift-operators
spec:
  channel: <ibm-mq-operator-channel>
  name: ibm-mq
  source: ibm-operator-catalog
  sourceNamespace: openshift-marketplace
```

Pour l'utilisation de `AllNamespaces` **InstallMode**, indiquez **openshift-operators** dans l'espace de nom. Sinon, indiquez l'espace de nom unique correspondant à l'utilisation de `Espace de nom unique` **InstallMode**. Notez que vous ne devez modifier que la zone **namespace**, en laissant la zone **sourceNamespace** comme c'est le cas.

c) Création de l'objet **Subscription**

```
oc apply -f mq-sub.yaml
```

8. Vérification du statut de l'opérateur

Une fois que l'installation de l'opérateur a abouti, l'état du pod s'affiche sous la forme *Exécution*. Pour l'utilisation de `AllNamespaces` **InstallMode**, indiquez **openshift-operators** comme espace de nom. Sinon, indiquez l'espace de nom unique correspondant à l'utilisation de `Espace de nom unique` **InstallMode**.

```
oc get pods -n <namespace_name>
```

Que faire ensuite

«Préparation de votre projet Red Hat OpenShift pour IBM MQ à l'aide de l'interface de ligne de commande Red Hat OpenShift», à la page 84

OpenShift CP4I *Désinstallation de IBM MQ Operator à l'aide de l'interface de ligne de commande Red Hat OpenShift*

Vous pouvez utiliser l'interface de ligne de commande Red Hat OpenShift pour désinstaller IBM MQ Operator de Red Hat OpenShift. Il existe des différences dans le processus de désinstallation, selon que IBM MQ Operator est installé dans un seul espace de nom, ou installé et disponible pour tous les espaces de nom sur le cluster.

Avant de commencer

Connectez-vous à votre cluster Red Hat OpenShift à l'aide de `oc login`.

Procédure

- Si IBM MQ Operator est installé dans un espace de nom unique, procédez comme suit :
  a) Vérifiez que vous êtes dans le bon projet :

  ```
  oc project <project_name>
  ```

  b) Affichez les gestionnaires de files d'attente installés dans le projet :

  ```
  oc get qmgr
  ```

  c) Supprimez un ou plusieurs gestionnaires de files d'attente :

  ```
  oc delete qmgr <qmgr_name>
  ```

  Notez que, bien que ces gestionnaires de files d'attente continuent à s'exécuter, ils risquent de ne pas fonctionner comme prévu sans IBM MQ Operator.

  d) Affichez les instances **ClusterServiceVersion** :

  ```
  oc get csv
  ```

  e) Supprimez le IBM MQ **ClusterServiceVersion**:

  ```
  oc delete csv <ibm_mq_csv_name>
  ```

  f) Affichez les abonnements :

  ```
  oc get subscription
  ```

  g) Supprimez tous les abonnements :

  ```
  oc delete subscription <ibm_mq_subscription_name>
  ```

  h) Facultatif : Si rien d'autre n'utilise les services communs, vous pouvez souhaiter désinstaller l'opérateur de services communs et supprimer le groupe d'opérateurs :

  a. Désinstallez l'opérateur de services communs, en suivant les instructions de la rubrique Désinstallation des services communs dans la documentation du produit IBM Cloud Pak foundational services.

  b. Affichez le groupe d'opérateurs :

  ```
  oc get operatorgroup
  ```

  c. Supprimez le groupe d'opérateurs :

  ```
  oc delete OperatorGroup <operator_group_name>
  ```

- Si IBM MQ Operator est installé et disponible pour tous les espaces de nom du cluster, procédez comme suit :
  a) Affichez tous les gestionnaires de files d'attente installés :

```
oc get qmgr -A
```

b) Supprimez un ou plusieurs gestionnaires de files d'attente :

```
oc delete qmgr <qmgr_name> -n <namespace_name>
```

Notez que, bien que ces gestionnaires de files d'attente continuent à s'exécuter, ils risquent de ne pas fonctionner comme prévu sans IBM MQ Operator.

c) Affichez les instances **ClusterServiceVersion** :

```
oc get csv -A
```

d) Supprimez le IBM MQ **ClusterServiceVersion** du cluster:

```
oc delete csv <ibm_mq_csv_name> -n openshift-operators
```

e) Affichez les abonnements :

```
oc get subscription -n openshift-operators
```

f) Supprimez les abonnements :

```
oc delete subscription <ibm_mq_subscription_name> -n openshift-operators
```

g) Si vous utilisez Red Hat OpenShift Container Platform 4.7, vous devrez peut-être supprimer manuellement le crochet de validation Web :

```
oc delete validatingwebhookconfiguration namespace.validator.queuemanagers.mq.ibm.com
```

h) Facultatif : Si rien d'autre n'utilise des services communs, vous pouvez souhaiter désinstaller l'opérateur de services communs :

Suivez les instructions de la rubrique Désinstallation des services communs dans la documentation du produit IBM Cloud Pak foundational services.

Installation d'IBM MQ Operator dans un environnement isolé physiquement

Ce tutoriel vous guide lors de l'installation de IBM MQ Operator dans un cluster Red Hat OpenShift qui n'a pas de connectivité Internet. Vous pouvez installer IBM MQ Operator dans un environnement isolé physiquement à l'aide d'un périphérique de stockage portable ou d'un serveur bastion.

Installation d'IBM MQ Operator dans un environnement isolé physiquement à l'aide d'un périphérique de stockage portable

Pour la procédure d'installation, reportez-vous à la rubrique Mirroring images with a portable storage device de la documentation IBM Cloud Pak for Integration. Si vous installez uniquement IBM MQ, remplacez toutes les occurrences des variables d'environnement suivantes par les valeurs indiquées ici:

```
export CASE_NAME=ibm-mq
export CASE_ARCHIVE_VERSION=version_number
export CASE_INVENTORY_SETUP=ibmMQOperator
```

où *version_number* est la version du dossier que vous souhaitez utiliser pour effectuer l'installation isolée physiquement. Pour obtenir la liste des versions de cas disponibles, voir https://github.com/IBM/cloud-pak/tree/master/repo/case/ibm-mq. Consultez la rubrique «Versions prises en charge pour IBM MQ Operator», à la page 7 pour déterminer le canal d'opérateur à sélectionner.

Installation d'IBM MQ Operator dans un environnement isolé physiquement à l'aide d'un serveur bastion

1. «Prérequis», à la page 69
2. «Préparation d'un registre Docker», à la page 69
3. «Préparation d'un hôte bastion», à la page 70
4. «Création de variables d'environnement pour le programme d'installation et l'inventaire des images», à la page 71
5. «Téléchargement du programme d'installation d'IBM MQ et de l'inventaire des images», à la page 71
6. «Connexion au cluster Red Hat OpenShift Container Platform en tant qu'administrateur de cluster», à la page 71
7. «Créez un espace de nom Kubernetes pour IBM MQ Operator», à la page 71
8. «Mise en miroir des images et configuration du cluster», à la page 72
9. «Installez IBM MQ Operator.», à la page 73
10. «Déploiement du gestionnaire de files d'attente IBM MQ», à la page 74

Prérequis

1. Un cluster Red Hat OpenShift Container Platform doit être installé. Pour prendre connaissance des versions d'Red Hat OpenShift Container Platform prises en charge, voir «Versions prises en charge pour IBM MQ Operator», à la page 7.
2. Un registre Docker doit être disponible. Pour plus d'informations, voir «Préparation d'un registre Docker», à la page 69.
3. Un serveur bastion doit être configuré. Pour plus d'informations, voir «Préparation d'un hôte bastion», à la page 70.

Préparation d'un registre Docker

Un registre Docker local est utilisé pour stocker toutes les images dans votre environnement local. Vous devez créer ce registre et vous assurer qu'il répond aux exigences suivantes :

- Il doit prendre en charge le schéma 2 du manifeste Docker version 2.
- Il doit prendre en charge les images à plusieurs architectures.
- Il doit être accessible depuis le serveur bastion et depuis vos noeuds de cluster Red Hat OpenShift Container Platform.
- Il doit avoir le nom d'utilisateur et le mot de passe d'un utilisateur pouvant écrire des données dans le registre cible à partir de l'hôte bastion.
- Possède le nom d'utilisateur et le mot de passe d'un utilisateur pouvant lire à partir du registre cible qui se trouve sur les nœuds de cluster Red Hat OpenShift.
- Il doit admettre les séparateurs de chemin dans les noms d'image.

Après avoir créé le registre Docker, vous devez le configurer :

1. Créez des espaces de nom de registre
   - `ibmcom` - Espace de nom permettant de stocker toutes les images provenant de l'espace de nom `dockerhub.io/ibmcom`.

     L'espace de nom `ibmcom` contient toutes les images IBM disponibles publiquement et dont l'extraction ne nécessite pas de données d'identification.
   - `cp` - Espace de nom permettant de stockage les images IBM qui proviennent du référentiel `cp.icr.io/cp`.

     L'espace de nom `cp` contient les images provenant du registre autorisé IBM dont l'extraction nécessite une autorisation d'utilisation du produit et des données d'identification. Pour obtenir

votre clé d'autorisation, connectez-vous à Mon IBM - Bibliothèque des logiciels de conteneur avec l'ID et le mot de passe IBM qui sont associés au logiciel autorisé. Dans la section **Clé d'autorisation**, sélectionnez **Copier la clé** pour copier la clé d'autorisation dans le presse-papiers, puis sauvegardez-la afin de l'utiliser au cours des étapes qui suivent.

- `opencloudio` - Espace de nom permettant de stockage les images qui proviennent de `quay.io/opencloudio`.

  L'espace de nom `opencloudio` contient une sélection d'images de composant open source IBM qui sont disponibles dans quay.io. Les images IBM Cloud Pak foundational services sont hébergées dans `opencloudio`.

2. Vérifiez que chaque espace-noms répond aux exigences suivantes :
   - Il prend en charge la création de référentiel automatique.
   - Il possède les données d'identification d'un utilisateur pouvant écrire des données dans des référentiels et créer des référentiels. L'hôte bastion utilise ces données d'identification.
   - Il possède les données d'identification d'un utilisateur qui peut lire tous les référentiels. La grappe Red Hat OpenShift Container Platform utilise ces données d'identification.

Préparation d'un hôte bastion

Préparez un hôte bastion pouvant accéder au cluster Red Hat OpenShift Container Platform, au registre Docker local et à Internet. L'hôte bastion doit se trouver sur une plateforme Linux for x86-64 sur laquelle est installé un système d'exploitation pris en charge par l'interface de ligne de commande IBM Cloud Pak et l'interface de ligne de commande Red Hat OpenShift Container Platform.

Effectuez les étapes suivantes sur votre noeud bastion :

1. Installez OpenSSL version 1.11.1 ou ultérieure.
2. Installez Docker ou Podman sur le noeud bastion.
   - Pour installer Docker, exécutez les commandes suivantes :

     ```
     yum check-update
     yum install docker
     ```

   - Pour installer Podman, voir Podman Installation Instructions.
3. Installez skopeo version 1.x.x sur le nœud bastion. Pour installer skopeo, exécutez les commandes suivantes :

   ```
   yum check-update
   yum install skopeo
   ```

4. Installez l'interface de ligne de commande IBM Cloud Pak. Installez la version la plus récente du fichier binaire pour votre plateforme. Pour plus d'informations, voir cloud-pak-cli.
   a. Téléchargez le fichier binaire.

      ```
      wget https://github.com/IBM/cloud-pak-cli/releases/download/vversion-number/binary-file-name
      ```

      Exemple :

      ```
      wget https://github.com/IBM/cloud-pak-cli/releases/latest/download/cloudctl-linux-amd64.tar.gz
      ```

   b. Procédez à l'extraction du fichier binaire.

      ```
      tar -xf binary-file-name
      ```

   c. Exécutez les commandes suivantes pour modifier et déplacer le fichier :

      ```
      chmod 755 file-name
      mv file-name /usr/local/bin/cloudctl
      ```

d. Vérifiez que `cloudctl` est installé :

```
cloudctl --help
```

5. Installez l'outil d'interface de ligne de commande oc Red Hat OpenShift Container Platform.

   Pour plus d'informations, voir Outils de l'interface de ligne de commande Red Hat OpenShift Container Platform.

6. Créez un répertoire qui servira de magasin hors ligne.

   Vous trouverez ci-après un exemple de répertoire. Cet exemple sera utilisé dans les étapes qui suivent.

   ```
   mkdir $HOME/offline
   ```

   Remarque : ce magasin hors ligne doit être persistant pour qu'il ne soit pas nécessaire de transférer des données plusieurs fois. La persistance permet également d'exécuter le processus de mise en miroir plusieurs fois ou selon une planification.

Création de variables d'environnement pour le programme d'installation et l'inventaire des images

Créez les variables d'environnement suivantes avec le nom de l'image du programme d'installation et l'inventaire des images:

```
export CASE_ARCHIVE_VERSION=version_number
export CASE_ARCHIVE=ibm-mq-$CASE_ARCHIVE_VERSION.tgz
export CASE_INVENTORY=ibmMQOperator
```

où *version_number* est la version du dossier que vous souhaitez utiliser pour effectuer l'installation isolée physiquement. Pour obtenir la liste des versions de cas disponibles, voir https://github.com/IBM/cloud-pak/tree/master/repo/case/ibm-mq. Consultez la rubrique Prise en charge de la version pour le IBM MQ Operator afin de déterminer le canal d'opérateur à choisir.

Téléchargement du programme d'installation d'IBM MQ et de l'inventaire des images

Téléchargez le programme d'installation `ibm-mq` et l'inventaire des images sur l'hôte bastion:

```
cloudctl case save \
    --case https://github.com/IBM/cloud-pak/raw/master/repo/case/ibm-mq/$CASE_ARCHIVE_VERSION/
$CASE_ARCHIVE \
    --outputdir $HOME/offline/
```

Connexion au cluster Red Hat OpenShift Container Platform en tant qu'administrateur de cluster

Voici un exemple de commande permettant de se connecter au cluster Red Hat OpenShift Container Platform :

```
oc login cluster_host:port --username=cluster_admin_user --password=cluster_admin_password
```

Créez un espace de nom Kubernetes pour IBM MQ Operator

Créez une variable d'environnement avec un espace de nom pour installer IBM MQ Operator, puis créez l'espace de nom:

```
export NAMESPACE=ibm-mq-test
oc create namespace ${NAMESPACE}
```

Mise en miroir des images et configuration du cluster

Procédez comme suit pour mettre les images en miroir et configurer votre cluster :

Remarque : N'utilisez pas le tilde entre des guillemets dans une quelconque commande. Par exemple, n'utilisez pas `args "--registry registry --user registry_userid --pass registry_password --inputDir ~/offline"`. Le tilde n'est pas développé et vos commandes peuvent échouer.

1. Stockez les données d'authentification pour tous les registres Docker source.

   Tous les services communs de la plateforme IBM Cloud, l'image IBM MQ Operator et l'image IBM MQ Advanced Developer sont stockés dans des registres publics ne nécessitant pas d'authentification. Toutefois, IBM MQ Advanced Server (autre que pour les développeurs), d'autres produits et des composants de tiers nécessitent un ou plusieurs registres authentifiés. Les registres suivants exigent une authentification :

   - `cp.icr.io`
   - `registry.redhat.io`
   - `registry.access.redhat.com`

   Pour plus d'informations sur ces registres, voir Créez des espaces de nom de registre.

   Vous devez exécuter la commande suivante afin de configurer les données d'identification pour tous les registres nécessitant une authentification. Exécutez la commande séparément pour chaque registre de ce type :

   ```
   cloudctl case launch \
   --case $HOME/offline/${CASE_ARCHIVE} \
   --inventory ${CASE_INVENTORY} \
   --action configure-creds-airgap \
   --namespace ${NAMESPACE} \
   --args "--registry registry --user registry_userid --pass registry_password --inputDir $HOME/
   offline"
   ```

   La commande stocke et met en cache les données d'identification de registre dans un fichier dans votre système de fichiers, dans l'emplacement `$HOME/.airgap/secrets`.

2. Créez des variables d'environnement contenant les informations de connexion au registre Docker local.

   ```
   export LOCAL_DOCKER_REGISTRY=IP_or_FQDN_of_local_docker_registry
   export LOCAL_DOCKER_USER=username
   export LOCAL_DOCKER_PASSWORD=password
   ```

   Remarque : le registre Docker utilise des ports standard tels que 80 ou 443. Si votre registre Docker utilise un port non standard, spécifiez le port à l'aide de la syntaxe *host*:*port*. Exemple :

   ```
   export LOCAL_DOCKER_REGISTRY=myregistry.local:5000
   ```

3. Configurez un secret d'authentification pour le registre Docker local.

   Remarque : cette étape ne doit être effectuée qu'une seule fois.

   ```
   cloudctl case launch \
   --case $HOME/offline/${CASE_ARCHIVE} \
   --inventory ${CASE_INVENTORY} \
   --action configure-creds-airgap \
   --namespace ${NAMESPACE} \
   --args "--registry ${LOCAL_DOCKER_REGISTRY} --user ${LOCAL_DOCKER_USER} --pass $
   {LOCAL_DOCKER_PASSWORD}"
   ```

   La commande stocke et met en cache les données d'identification de registre dans un fichier dans votre système de fichiers, dans l'emplacement `$HOME/.airgap/secrets`.

4. Configurez un secret d'extraction d'image globale et la ressource **ImageContentSourcePolicy**.

   a. Vérifiez si un redémarrage du noeud est requis.

- Dans Red Hat OpenShift Container Platform version 4.4 et versions ultérieures, et sur une nouvelle installation d' IBM MQ Operator à l'aide d'airgap, cette étape redémarre tous les noeuds de cluster. Il se peut que les ressources du cluster ne soient pas disponibles tant que le nouveau secret d'extraction n'est pas appliqué.
- Dans IBM MQ Operator 1.8, CASE est mis à jour pour inclure une source de mise en miroir supplémentaire pour les images. Par conséquent, lorsque vous effectuez une mise à niveau depuis des versions précédentes de IBM MQ Operator vers la version 1.8 ou une version ultérieure, un redémarrage de noeud est déclenché.
- Pour vérifier si cette étape nécessite un redémarrage du noeud, ajoutez l'option `--dry-run` au code de cette étape. Cela génère le dernier **ImageContentSourcePolicy** et l'affiche dans la fenêtre de la console (**stdout**). Si cette **ImageContentSourcePolicy** diffère du cluster configuré **ImageContentSourcePolicy**, un redémarrage est effectué.

```
cloudctl case launch \
--case $HOME/offline/${CASE_ARCHIVE} \
--inventory ${CASE_INVENTORY} \
--action configure-cluster-airgap \
--namespace ${NAMESPACE} \
--args "--registry ${LOCAL_DOCKER_REGISTRY} --user ${LOCAL_DOCKER_USER} --pass $
{LOCAL_DOCKER_PASSWORD} --inputDir $HOME/offline --dryRun"
```

b. Pour configurer le secret d'extraction d'image global et **ImageContentSourcePolicy**, exécutez le code de cette étape sans l'option `--dry-run` :

```
cloudctl case launch \
--case $HOME/offline/${CASE_ARCHIVE} \
--inventory ${CASE_INVENTORY} \
--action configure-cluster-airgap \
--namespace ${NAMESPACE} \
--args "--registry ${LOCAL_DOCKER_REGISTRY} --user ${LOCAL_DOCKER_USER} --pass $
{LOCAL_DOCKER_PASSWORD} --inputDir $HOME/offline"
```

5. Vérifiez que la ressource **ImageContentSourcePolicy** a été créée.

```
oc get imageContentSourcePolicy
```

6. Facultatif : si vous utilisez un registre non sécurisé, vous devez ajouter le registre local à la liste **insecureRegistries** du cluster.

```
oc patch image.config.openshift.io/cluster --type=merge -p '{"spec":{"registrySources":
{"insecureRegistries":["'${LOCAL_DOCKER_REGISTRY}'"]}}}'
```

7. Vérifiez le statut de votre noeud de cluster.

```
oc get nodes
```

Une fois que **imageContentsourcePolicy** et le secret d'extraction d'image globale sont appliqués, le statut du noeud peut être **Ready**, **Scheduling** ou **Disabled**. Attendez que tous les noeuds affichent le statut **Ready**.

8. Mettez les images en miroir dans le registre local.

```
cloudctl case launch \
--case $HOME/offline/${CASE_ARCHIVE} \
--inventory ${CASE_INVENTORY} \
--action mirror-images \
--namespace ${NAMESPACE} \
--args "--registry ${LOCAL_DOCKER_REGISTRY} --user ${LOCAL_DOCKER_USER} --pass $
{LOCAL_DOCKER_PASSWORD} --inputDir $HOME/offline"
```

Installez IBM MQ Operator.

1. Connectez-vous à votre console Web de cluster Red Hat OpenShift.
2. Créez une source de catalogue. Utilisez le terminal qui a exécuté les étapes précédentes.

```
cloudctl case launch \
--case $HOME/offline/${CASE_ARCHIVE} \
--inventory ${CASE_INVENTORY} \
--action install-catalog \
--namespace ${NAMESPACE} \
--args "--registry ${LOCAL_DOCKER_REGISTRY} --recursive"
```

3. Vérifiez que la ressource **CatalogSource** est créée pour l'opérateur du programme d'installation des services communs.

```
oc get pods -n openshift-marketplace
oc get catalogsource -n openshift-marketplace
```

4. Installez IBM MQ Operator à l'aide d'OLM.
   a. Depuis le panneau de navigation, cliquez sur **Operators** > **OperatorHub**.

      La page **OperatorHub** s'affiche.
   b. Dans la zone **Tous les articles**, entrez `IBM MQ`.

      L'entrée de catalogue IBM MQ est affichée.
   c. Sélectionnez **IBM MQ**.

      La fenêtre **IBM MQ** s'ouvre.
   d. Cliquez sur **Install**.

      La page **Create Operator Subscription** s'affiche.
   e. Consultez la rubrique «Versions prises en charge pour IBM MQ Operator», à la page 7 pour déterminer le canal d'opérateur à sélectionner.
   f. Dans la zone **Installation Mode**, choisissez de procéder à l'installation dans l'espace de nom spécifique que vous avez créé ou dans le cluster.
   g. Cliquez sur **Subscribe**.

      IBM MQ est ajouté dans la page **Installed Operators**.
   h. Vérifiez le statut de l'opérateur dans la page **Installed Operators**. Le statut devient **Succeeded** une fois l'installation terminée.

Déploiement du gestionnaire de files d'attente IBM MQ

Pour créer un gestionnaire de files d'attente sous l'opérateur installé, voir «Déploiement et configuration de gestionnaires de files d'attente à l'aide de IBM MQ Operator», à la page 83.

Tâches associées

«Préparation de la mise à niveau du IBM MQ Operator ou du gestionnaire de files d'attente dans un environnement airgap», à la page 75
Dans un cluster Red Hat OpenShift qui n'a pas de connectivité Internet, vous devez effectuer des étapes préparatoires avant de mettre à niveau le IBM MQ Operator.

OpenShift CP4I Mise à niveau de IBM MQ Operator et des gestionnaires de files d'attente

La mise à niveau d'IBM MQ Operator vous permet de mettre à niveau vos gestionnaires de files d'attente.

Procédure

- «Mise à niveau de IBM MQ Operator à l'aide de la console Web Red Hat OpenShift», à la page 78.
- «Mise à niveau de IBM MQ Operator à l'aide de l'interface CLI Red Hat OpenShift», à la page 79.
- «Mise à niveau d'un gestionnaire de files d'attente IBM MQ à l'aide de la console Web Red Hat OpenShift», à la page 81.

- «Mise à niveau d'un gestionnaire de files d'attente IBM MQ à l'aide de Red Hat OpenShift CLI», à la page 81.

OpenShift CP4I Linux Préparation de la mise à niveau du IBM MQ Operator ou du gestionnaire de files d'attente dans un environnement airgap

Dans un cluster Red Hat OpenShift qui n'a pas de connectivité Internet, vous devez effectuer des étapes préparatoires avant de mettre à niveau le IBM MQ Operator.

Avant de commencer

Cette rubrique suppose que vous avez déjà configuré un registre d'images local dans lequel les images IBM Cloud Pak for Integration précédemment publiées sont mises en miroir.

Pourquoi et quand exécuter cette tâche

Pour pouvoir mettre à niveau le IBM MQ Operator ou le gestionnaire de files d'attente dans un environnement airgap, vous devez mettre en miroir les dernières images IBM Cloud Pak for Integration .

Notez que les quatre premières étapes de cette tâche sont les mêmes que celles que vous effectuez lorsque «Installation d'IBM MQ Operator dans un environnement isolé physiquement», à la page 68.

Procédure

1. Créez des variables d'environnement pour le programme d'installation et l'inventaire des images.

   Créez les variables d'environnement suivantes avec le nom de l'image du programme d'installation et l'inventaire des images:

   ```
   export CASE_ARCHIVE_VERSION=version_number
   export CASE_ARCHIVE=ibm-mq-$CASE_ARCHIVE_VERSION.tgz
   export CASE_INVENTORY=ibmMQOperator
   ```

   où *version_number* est la version du dossier que vous souhaitez utiliser pour effectuer l'installation isolée physiquement. Pour obtenir la liste des versions de cas disponibles, voir https://github.com/IBM/cloud-pak/tree/master/repo/case/ibm-mq. Consultez la rubrique Prise en charge de la version pour le IBM MQ Operator afin de déterminer le canal d'opérateur à choisir.
2. Téléchargez le programme d'installation IBM MQ et l'inventaire des images.

   Téléchargez le programme d'installation `ibm-mq` et l'inventaire des images sur l'hôte bastion:

   ```
   cloudctl case save \
       --case https://github.com/IBM/cloud-pak/raw/master/repo/case/ibm-mq/
   $CASE_ARCHIVE_VERSION/$CASE_ARCHIVE \
       --outputdir $HOME/offline/
   ```
3. Connectez-vous à la grappe Red Hat OpenShift Container Platform en tant qu'administrateur de la grappe.

   Voici un exemple de commande permettant de se connecter au cluster Red Hat OpenShift Container Platform :

   ```
   oc login cluster_host:port --username=cluster_admin_user --password=cluster_admin_password
   ```
4. Mettez les images en miroir et configurez le cluster.

   Procédez comme suit pour mettre les images en miroir et configurer votre cluster :

   Remarque : N'utilisez pas le tilde entre des guillemets dans une quelconque commande. Par exemple, n'utilisez pas `args "--registry registry --user registry_userid --pass registry_password --inputDir ~/offline"`. Le tilde n'est pas développé et vos commandes peuvent échouer.

   a. Stockez les données d'authentification pour tous les registres Docker source.

Tous les services communs de la plateforme IBM Cloud, l'image IBM MQ Operator et l'image IBM MQ Advanced Developer sont stockés dans des registres publics ne nécessitant pas d'authentification. Toutefois, IBM MQ Advanced Server (autre que pour les développeurs), d'autres produits et des composants de tiers nécessitent un ou plusieurs registres authentifiés. Les registres suivants exigent une authentification :

- `cp.icr.io`
- `registry.redhat.io`
- `registry.access.redhat.com`

Pour plus d'informations sur ces registres, voir Créez des espaces de nom de registre.

Vous devez exécuter la commande suivante afin de configurer les données d'identification pour tous les registres nécessitant une authentification. Exécutez la commande séparément pour chaque registre de ce type :

```
cloudctl case launch \
--case $HOME/offline/${CASE_ARCHIVE} \
--inventory ${CASE_INVENTORY} \
--action configure-creds-airgap \
--namespace ${NAMESPACE} \
--args "--registry registry --user registry_userid --pass registry_password --inputDir
$HOME/offline"
```

La commande stocke et met en cache les données d'identification de registre dans un fichier dans votre système de fichiers, dans l'emplacement `$HOME/.airgap/secrets`.

b. Créez des variables d'environnement contenant les informations de connexion au registre Docker local.

```
export LOCAL_DOCKER_REGISTRY=IP_or_FQDN_of_local_docker_registry
export LOCAL_DOCKER_USER=username
export LOCAL_DOCKER_PASSWORD=password
```

Remarque : le registre Docker utilise des ports standard tels que 80 ou 443. Si votre registre Docker utilise un port non standard, spécifiez le port à l'aide de la syntaxe *host*:*port*. Exemple :

```
export LOCAL_DOCKER_REGISTRY=myregistry.local:5000
```

c. Configurez un secret d'authentification pour le registre Docker local.

Remarque : cette étape ne doit être effectuée qu'une seule fois.

```
cloudctl case launch \
--case $HOME/offline/${CASE_ARCHIVE} \
--inventory ${CASE_INVENTORY} \
--action configure-creds-airgap \
--namespace ${NAMESPACE} \
--args "--registry ${LOCAL_DOCKER_REGISTRY} --user ${LOCAL_DOCKER_USER} --pass $
{LOCAL_DOCKER_PASSWORD}"
```

La commande stocke et met en cache les données d'identification de registre dans un fichier dans votre système de fichiers, dans l'emplacement `$HOME/.airgap/secrets`.

d. Configurez un secret d'extraction d'image globale et la ressource **`ImageContentSourcePolicy`**.

i) Vérifiez si un redémarrage du noeud est requis.

- Dans Red Hat OpenShift Container Platform version 4.4 et versions ultérieures, et sur une nouvelle installation d' IBM MQ Operator à l'aide d'airgap, cette étape redémarre tous les noeuds de cluster. Il se peut que les ressources du cluster ne soient pas disponibles tant que le nouveau secret d'extraction n'est pas appliqué.
- Dans IBM MQ Operator 1.8, CASE est mis à jour pour inclure une source de mise en miroir supplémentaire pour les images. Par conséquent, lorsque vous effectuez une mise à niveau depuis des versions précédentes de IBM MQ Operator vers la version 1.8 ou une version ultérieure, un redémarrage de noeud est déclenché.

- Pour vérifier si cette étape nécessite un redémarrage du noeud, ajoutez l'option `--dry-run` au code de cette étape. Cela génère le dernier **ImageContentSourcePolicy** et l'affiche dans la fenêtre de la console (**stdout**). Si cette **ImageContentSourcePolicy** diffère du cluster configuré **ImageContentSourcePolicy**, un redémarrage est effectué.

```
cloudctl case launch \
--case $HOME/offline/${CASE_ARCHIVE} \
--inventory ${CASE_INVENTORY} \
--action configure-cluster-airgap \
--namespace ${NAMESPACE} \
--args "--registry ${LOCAL_DOCKER_REGISTRY} --user ${LOCAL_DOCKER_USER} --pass $
{LOCAL_DOCKER_PASSWORD} --inputDir $HOME/offline --dryRun"
```

ii) Pour configurer le secret d'extraction d'image global et **ImageContentSourcePolicy**, exécutez le code de cette étape sans l'option `--dry-run` :

```
cloudctl case launch \
--case $HOME/offline/${CASE_ARCHIVE} \
--inventory ${CASE_INVENTORY} \
--action configure-cluster-airgap \
--namespace ${NAMESPACE} \
--args "--registry ${LOCAL_DOCKER_REGISTRY} --user ${LOCAL_DOCKER_USER} --pass $
{LOCAL_DOCKER_PASSWORD} --inputDir $HOME/offline"
```

e. Vérifiez que la ressource **ImageContentSourcePolicy** a été créée.

```
oc get imageContentSourcePolicy
```

f. Facultatif : si vous utilisez un registre non sécurisé, vous devez ajouter le registre local à la liste **insecureRegistries** du cluster.

```
oc patch image.config.openshift.io/cluster --type=merge -p '{"spec":{"registrySources":
{"insecureRegistries":["'${LOCAL_DOCKER_REGISTRY}'"]}}}'
```

g. Vérifiez le statut de votre noeud de cluster.

```
oc get nodes
```

Une fois que **imageContentsourcePolicy** et le secret d'extraction d'image globale sont appliqués, le statut du noeud peut être **Ready**, **Scheduling** ou **Disabled**. Attendez que tous les noeuds affichent le statut **Ready**.

h. Mettez les images en miroir dans le registre local.

```
cloudctl case launch \
--case $HOME/offline/${CASE_ARCHIVE} \
--inventory ${CASE_INVENTORY} \
--action mirror-images \
--namespace ${NAMESPACE} \
--args "--registry ${LOCAL_DOCKER_REGISTRY} --user ${LOCAL_DOCKER_USER} --pass $
{LOCAL_DOCKER_PASSWORD} --inputDir $HOME/offline"
```

5. Mettez à niveau la source de catalogue.

Utilisez le terminal qui a exécuté les étapes précédentes.

```
cloudctl case launch \
--case $HOME/offline/${CASE_ARCHIVE} \
--inventory ${CASE_INVENTORY} \
--action install-catalog \
--namespace ${NAMESPACE} \
--args "--registry ${LOCAL_DOCKER_REGISTRY} --recursive"
```

Que faire ensuite

Vous êtes maintenant prêt à mettre à niveau le IBM MQ Operator et le gestionnaire de files d'attente en exécutant l'une des tâches suivantes:

- «Mise à niveau de IBM MQ Operator à l'aide de la console Web Red Hat OpenShift», à la page 78

- «Mise à niveau de IBM MQ Operator à l'aide de l'interface CLI Red Hat OpenShift», à la page 79
- «Mise à niveau d'un gestionnaire de files d'attente IBM MQ à l'aide de la console Web Red Hat OpenShift», à la page 81
- «Mise à niveau d'un gestionnaire de files d'attente IBM MQ à l'aide de Red Hat OpenShift CLI», à la page 81
- «Mise à niveau d'un gestionnaire de files d'attente IBM MQ dans Red Hat OpenShift à l'aide de la plateforme Navigator», à la page 82

OpenShift CP4I Mise à niveau de IBM MQ Operator à l'aide de la console Web Red Hat OpenShift

IBM MQ Operator peut être mis à niveau à l'aide d'Operator Hub.

Avant de commencer

Connectez-vous à votre console Web de cluster Red Hat OpenShift.

Avant de pouvoir mettre à niveau le IBM MQ Operator dans un environnement airgap, vous devez reproduire les dernières images IBM Cloud Pak for Integration . Voir Préparation de la mise à niveau du IBM MQ Operator ou du gestionnaire de files d'attente dans un environnement airgap.

Procédure

1. Consultez la rubrique «Versions prises en charge pour IBM MQ Operator», à la page 7 pour déterminer le canal d'opérateur vers lequel vous souhaitez effectuer la mise à niveau.
2. Facultatif : Si vous effectuez une mise à niveau à partir d'une version de IBM MQ Operator antérieure à 1.5 à IBM MQ Operator 1.5 ou ultérieure, vous devez d'abord mettre à niveau la version de IBM Cloud Pak foundational services.

   Pour plus d'informations, voir «Mise à niveau de IBM Cloud Pak foundational services à l'aide de la console Web Red Hat OpenShift», à la page 79.
3. Mettez à niveau IBM MQ Operator. Les nouvelles versions principales ou mineures de IBM MQ Operator sont livrées via de nouveaux canaux d'abonnement. Pour mettre votre opérateur à niveau vers une nouvelle version principale ou mineure, vous devez mettre à jour le canal sélectionné dans votre abonnement IBM MQ Operator.
   a) Depuis le panneau de navigation, cliquez sur **Operators** > **Installed Operators**.

      Tous les opérateurs installés dans le projet spécifié sont affichés.
   b) Sélectionnez **IBM MQ Operator**.
   c) Accédez à l'onglet **Subscription**.
   d) Cliquez sur **Channel**.

      La fenêtre **Change Subscription Update Channel** s'ouvre.
   e) Sélectionnez le canal de votre choix et cliquez sur **Save**.

      L'opérateur est mis à niveau avec la version la plus récente disponible sur le nouveau canal. Voir «Versions prises en charge pour IBM MQ Operator», à la page 7.

Que faire ensuite

Si vous avez effectué une mise à niveau vers IBM Cloud Pak foundational services 3.7, les gestionnaires de files d'attente qui utilisent une licence IBM Cloud Pak for Integration doivent être mis à niveau ou redémarrés. Pour plus d'informations sur cette procédure, voir «Mise à niveau d'un gestionnaire de files d'attente IBM MQ à l'aide de la console Web Red Hat OpenShift», à la page 81.

OpenShift CP4I *Mise à niveau de IBM Cloud Pak foundational services à l'aide de la console Web Red Hat OpenShift*

Si vous effectuez une mise à niveau à partir d'une version de IBM MQ Operator antérieure à 1.5 à IBM MQ Operator 1.5 ou ultérieure, vous devez d'abord mettre à niveau la version de IBM Cloud Pak foundational services.

Avant de commencer

Remarque : Vous devez effectuer cette tâche uniquement si vous effectuez une mise à niveau à partir d'une version de IBM MQ Operator antérieure à 1.5 à IBM MQ Operator 1.5 ou ultérieure.

CP4I Si vous disposez de gestionnaires de files d'attente qui utilisent une licence IBM Cloud Pak for Integration, après cette mise à niveau, un redémarrage du gestionnaire de files d'attente est requis pour accéder à la console Web, et vous observerez également d'autres erreurs de connexion à la console Web. Vous pouvez corriger ces erreurs en mettant à niveau la dernière valeur de `.spec.version` pour la version IBM MQ choisie, une fois la mise à niveau de l'opérateur terminée.

CP4I Si vous disposez de gestionnaires de files d'attente existants et que vous utilisez le tableau de bord des opérations IBM Cloud Pak for Integration, regardez «Déploiement ou mise à niveau de IBM MQ 9.2.2 ou 9.2.3 avec l'intégration du tableau de bord des opérations dans IBM Cloud Pak for Integration 2021.4», à la page 116 avant la mise à niveau.

Procédure

1. Connectez-vous à votre console Web de cluster Red Hat OpenShift.
2. Depuis le panneau de navigation, cliquez sur **Operators** > **Installed Operators**.
   Tous les opérateurs installés dans le projet spécifié sont affichés.
3. Sélectionnez **IBM Cloud Pak foundational services Operator**. Notez qu'avant la version 3.7, il s'appelait **IBM Common Services Operator**.
4. Accédez à l'onglet **Abonnement** .
5. Cliquez sur **Canal**.
   La fenêtre **Change Subscription Update Channel** s'ouvre.
6. Sélectionnez le canal **v3**, puis cliquez sur **Sauvegarder**.
   L'opérateur IBM Cloud Pak foundational services met à niveau la dernière version disponible pour le nouveau canal. Voir «Versions prises en charge pour IBM MQ Operator», à la page 7.

Que faire ensuite

Vous êtes maintenant prêt à Mettre à niveau IBM MQ Operator.

OpenShift CP4I Mise à niveau de IBM MQ Operator à l'aide de l'interface CLI Red Hat OpenShift

IBM MQ Operator peut être mis à niveau à partir de la ligne de commande.

Avant de commencer

Connectez-vous à votre cluster avec **`cloudctl login`** (pour IBM Cloud Pak for Integration) ou **`oc login`**.

Avant de pouvoir mettre à niveau le IBM MQ Operator dans un environnement airgap, vous devez reproduire les dernières images IBM Cloud Pak for Integration . Voir Préparation de la mise à niveau du IBM MQ Operator ou du gestionnaire de files d'attente dans un environnement airgap.

Procédure

1. Consultez la rubrique «Versions prises en charge pour IBM MQ Operator», à la page 7 pour déterminer le canal d'opérateur vers lequel vous souhaitez effectuer la mise à niveau.
2. Facultatif : Si vous effectuez une mise à niveau à partir d'une version de IBM MQ Operator antérieure à 1.5 à IBM MQ Operator 1.5 ou ultérieure, vous devez d'abord mettre à niveau la version de IBM Cloud Pak foundational services.

   Pour plus d'informations, voir «Mise à niveau de IBM Cloud Pak foundational services à l'aide de l'interface CLI Red Hat OpenShift», à la page 80.
3. Mettez à niveau IBM MQ Operator. De nouvelles versions IBM MQ Operator majeures/mineures sont distribuées via les nouveaux canaux d'abonnement. Pour mettre à niveau votre opérateur vers une nouvelle version majeure/mineure, vous devez mettre à jour le canal sélectionné dans votre abonnement IBM MQ Operator.

   a) Vérifiez que le canal de mise à niveau IBM MQ Operator requis est disponible.

   ```
   oc get packagemanifest ibm-mq -o=jsonpath='{.status.channels[*].name}'
   ```

   b) Utilisez le correctif de `Subscription` pour passer au canal de mise à jour souhaité (où *VX.O* est le canal de mise à jour souhaité identifié à l'étape précédente.

   ```
   oc patch subscription ibm-mq --patch '{"spec":{"channel":"vX.Y"}}' --type=merge
   ```

Que faire ensuite

Si vous avez effectué une mise à niveau vers IBM Cloud Pak foundational services 3.7, les gestionnaires de files d'attente qui utilisent une licence IBM Cloud Pak for Integration doivent être mis à niveau ou redémarrés. Pour plus d'informations sur cette procédure, voir «Mise à niveau d'un gestionnaire de files d'attente IBM MQ à l'aide de Red Hat OpenShift CLI», à la page 81.

OpenShift CP4I *Mise à niveau de IBM Cloud Pak foundational services à l'aide de l'interface CLI Red Hat OpenShift*

Si vous effectuez une mise à niveau à partir d'une version de IBM MQ Operator antérieure à 1.5 à IBM MQ Operator 1.5 ou ultérieure, vous devez d'abord mettre à niveau la version de IBM Cloud Pak foundational services.

Avant de commencer

Remarque : Vous devez effectuer cette tâche uniquement si vous effectuez une mise à niveau à partir d'une version de IBM MQ Operator antérieure à 1.5 à IBM MQ Operator 1.5 ou ultérieure.

CP4I Si l'un de vos gestionnaires de files d'attente utilise une licence IBM Cloud Pak for Integration, après cette mise à niveau, un redémarrage de ce gestionnaire de files d'attente est requis pour accéder à la console Web et d'autres erreurs sont consignées dans la console Web. Vous pouvez corriger ces erreurs en mettant à niveau la dernière valeur de `.spec.version` pour la version IBM MQ choisie, une fois la mise à niveau de l'opérateur terminée.

CP4I Si vous disposez de gestionnaires de files d'attente existants et que vous utilisez le tableau de bord des opérations IBM Cloud Pak for Integration, regardez «Déploiement ou mise à niveau de IBM MQ 9.2.2 ou 9.2.3 avec l'intégration du tableau de bord des opérations dans IBM Cloud Pak for Integration 2021.4», à la page 116 avant la mise à niveau.

Procédure

1. Connectez-vous à votre cluster avec **`cloudctl login`** (pour IBM Cloud Pak for Integration) ou **`oc login`**.
2. Vérifiez que le canal de mise à niveau v3 IBM Cloud Pak foundational services est disponible.

```
oc get packagemanifest -n ibm-common-services ibm-common-service-operator
-o=jsonpath='{.status.channels[*].name}'
```

3. Utiliser le correctif `Subscription` pour passer au canal de mise à jour souhaité : v3

```
oc patch subscription ibm-common-service-operator --patch '{"spec":{"channel":"v3"}}' --
type=merge
```

Que faire ensuite

Vous êtes maintenant prêt à Mettre à niveau IBM MQ Operator.

OpenShift CP4I Mise à niveau d'un gestionnaire de files d'attente IBM MQ à l'aide de la console Web Red Hat OpenShift

Un gestionnaire de files d'attente IBM MQ, déployé à l'aide de IBM MQ Operator, peut être mis à niveau dans Red Hat OpenShift à l'aide de l'opérateur Hub.

Avant de commencer

- Connectez-vous à votre console Web de cluster Red Hat OpenShift.
- Vérifiez que IBM MQ Operator utilise le canal de mise à jour souhaité. Voir «Mise à niveau de IBM MQ Operator et des gestionnaires de files d'attente», à la page 74.

Avant de pouvoir mettre à niveau le gestionnaire de files d'attente dans un environnement airgap, vous devez mettre en miroir les dernières images IBM Cloud Pak for Integration . Voir Préparation de la mise à niveau du IBM MQ Operator ou du gestionnaire de files d'attente dans un environnement airgap.

Procédure

1. Depuis le panneau de navigation, cliquez sur **Operators** > **Installed Operators**.
   Tous les opérateurs installés dans le projet spécifié sont affichés.
2. Sélectionnez **IBM MQ Operator**.
   La fenêtre **IBM MQ Operator** s'affiche.
3. Accédez à l'onglet **Queue Manager** .
   La fenêtre **Queue Manager Details** s'affiche.
4. Sélectionnez le gestionnaire de files d'attente à mettre à niveau.
5. Accédez à l'onglet YAML.
6. Mettez à jour les zones suivantes, le cas échéant, pour qu'elles correspondent à la mise à niveau de la version du gestionnaire de files d'attente IBM MQ souhaitée.
   - spec.version
   - spec.license.licence

   Voir «Versions prises en charge pour IBM MQ Operator», à la page 7 pour un mappage des canaux vers les versions IBM MQ Operator et les versions du gestionnaire de files d'attente IBM MQ.
7. Sauvegardez le gestionnaire de files d'attente mis à jour YAML.

OpenShift CP4I Mise à niveau d'un gestionnaire de files d'attente IBM MQ à l'aide de Red Hat OpenShift CLI

Un gestionnaire de files d'attente IBM MQ, déployé à l'aide de IBM MQ Operator, peut être mis à niveau dans Red Hat OpenShift à l'aide de la ligne de commande.

Avant de commencer

Vous devez être administrateur de cluster pour effectuer les étapes suivantes.

- Connectez-vous à l'interface de ligne de commande Red Hat OpenShift à l'aide de `oc login`.
- Vérifiez que IBM MQ Operator utilise le canal de mise à jour souhaité. Voir «Mise à niveau de IBM MQ Operator et des gestionnaires de files d'attente», à la page 74.

Avant de pouvoir mettre à niveau le gestionnaire de files d'attente dans un environnement airgap, vous devez mettre en miroir les dernières images IBM Cloud Pak for Integration . Voir Préparation de la mise à niveau du IBM MQ Operator ou du gestionnaire de files d'attente dans un environnement airgap.

Procédure

Editez la ressource **QueueManager** pour mettre à jour les zones suivantes, le cas échéant, pour qu'elles correspondent à la mise à niveau de la version du gestionnaire de files d'attente IBM MQ souhaitée.

- spec.version
- spec.license.licence

Voir «Versions prises en charge pour IBM MQ Operator», à la page 7 pour un mappage des canaux vers les versions IBM MQ Operator et les versions du gestionnaire de files d'attente IBM MQ.

Utilisez la commande suivante :

```
oc edit queuemanager my_qmgr
```

où *my_qmgr* est le nom de la ressource QueueManager que vous souhaitez mettre à niveau.

CP4I Mise à niveau d'un gestionnaire de files d'attente IBM MQ dans Red Hat OpenShift à l'aide de la plateforme Navigator

Un gestionnaire de files d'attente IBM MQ, déployé à l'aide de IBM MQ Operator, peut être mis à niveau dans Red Hat OpenShift à l'aide de IBM Cloud Pak for Integration Platform Navigator.

Avant de commencer

- Connectez-vous à IBM Cloud Pak for Integration Platform Navigator dans l'espace de nom contenant le gestionnaire de files d'attente que vous souhaitez mettre à niveau.
- Vérifiez que IBM MQ Operator utilise le canal de mise à jour souhaité. Voir «Mise à niveau de IBM MQ Operator et des gestionnaires de files d'attente», à la page 74.

Avant de pouvoir mettre à niveau le gestionnaire de files d'attente dans un environnement airgap, vous devez mettre en miroir les dernières images IBM Cloud Pak for Integration . Voir Préparation de la mise à niveau du IBM MQ Operator ou du gestionnaire de files d'attente dans un environnement airgap.

Procédure

1. A partir de la page d'accueil IBM Cloud Pak for Integration Platform Navigator, cliquez sur l'onglet **Runtimes**.
2. Les gestionnaires de files d'attente dont les mises à niveau sont disponibles ont un **i** bleu en regard de la **version**. Cliquez sur la lettre **i** pour afficher la **nouvelle version disponible**.
3. Cliquez sur les trois points à l'extrême droite du gestionnaire de files d'attente que vous souhaitez mettre à niveau, puis cliquez sur **Change version**.
4. Sous **Select a new channel or version**, sélectionnez la version de mise à niveau requise.
5. Cliquez sur **Change version**.

Résultats

Le gestionnaire de files d'attente est mis à niveau.

OpenShift CP4I Déploiement et configuration de gestionnaires de files d'attente à l'aide de IBM MQ Operator

IBM MQ 9.1.5 et versions ultérieures sont déployés sur Red Hat OpenShift à l'aide de IBM MQ Operator.

Pourquoi et quand exécuter cette tâche

Procédure

- «Préparation de votre projet Red Hat OpenShift pour IBM MQ», à la page 83.
- «Déploiement d'un gestionnaire de files d'attente sur un cluster Red Hat OpenShift Container Platform», à la page 85.

OpenShift CP4I Préparation de votre projet Red Hat OpenShift pour IBM MQ

Préparez votre cluster Red Hat OpenShift Container Platform pour qu'il soit prêt à déployer un gestionnaire de files d'attente.

Procédure

- «Préparation de votre projet Red Hat OpenShift pour IBM MQ à l'aide de la console Web Red Hat OpenShift», à la page 83.
- «Préparation de votre projet Red Hat OpenShift pour IBM MQ à l'aide de l'interface de ligne de commande Red Hat OpenShift», à la page 84.

Tâches associées

«Déploiement d'un gestionnaire de files d'attente sur un cluster Red Hat OpenShift Container Platform», à la page 85
Utilisez la ressource personnalisée QueueManager pour déployer un gestionnaire de files d'attente sur un cluster Red Hat OpenShift Container Platform.

OpenShift CP4I *Préparation de votre projet Red Hat OpenShift pour IBM MQ à l'aide de la console Web Red Hat OpenShift*

Préparez votre cluster Red Hat OpenShift Container Platform pour qu'il puisse déployer un gestionnaire de files d'attente à l'aide d'IBM MQ Operator. Cette tâche doit être effectuée par un administrateur de projet.

Avant de commencer

Remarque : Si vous prévoyez d'utiliser IBM MQ dans un projet dans lequel d'autres composants IBM Cloud Pak for Integration sont déjà installés, il n'est pas nécessaire de suivre les instructions ci-dessous.

Connectez-vous à votre console Web de cluster Red Hat OpenShift.

Pourquoi et quand exécuter cette tâche

Les images de IBM MQ Operator sont extraites depuis un registre de conteneurs qui effectue un contrôle des autorisations de licence. Ce contrôle requiert une clé d'autorisation qui est stockée dans un secret d'extraction `docker-registry`. Si vous ne possédez pas encore de clé d'autorisation, suivez les instructions ci-après pour en obtenir une et créer un secret d'extraction.

Procédure

1. Obtenez la clé d'autorisation qui est affectée à votre ID.

a) Connectez-vous à Mon IBM - Bibliothèque des logiciels de conteneur avec l'ID et le mot de passe IBM associés au logiciel autorisé.

b) Dans la section **Clé d'autorisation**, cliquez sur **Copier la clé** pour copier la clé d'autorisation dans le presse-papiers.

2. Créez un secret contenant votre clé d'autorisation dans le projet dans lequel vous voulez déployer votre gestionnaire de files d'attente.

a) Depuis le panneau de navigation, cliquez sur **Workloads** > **Secret**.

La page Secrets s'affiche.

b) Dans la liste déroulante **Project**, sélectionnez le projet dans lequel vous voulez installer IBM MQ.

c) Cliquez sur le bouton **Create** et sélectionnez **Image Pull Secret**.

d) Dans la zone **Name**, entrez `ibm-entitlement-key`.

e) Dans la zone **Registry Server Address**, entrez `cp.icr.io`.

f) Dans la zone **Username**, entrez `cp`.

g) Dans la zone **Password**, entrez la clé d'autorisation que vous avez copiée à l'étape précédente.

h) Dans la zone **Email**, entrez l'IBMid associé au logiciel autorisé.

Que faire ensuite

«Déploiement d'un gestionnaire de files d'attente à l'aide de la console Web Red Hat OpenShift», à la page 86

OpenShift CP4I *Préparation de votre projet Red Hat OpenShift pour IBM MQ à l'aide de l'interface de ligne de commande Red Hat OpenShift*

Préparez votre cluster Red Hat OpenShift Container Platform pour qu'il puisse déployer un gestionnaire de files d'attente à l'aide d'IBM MQ Operator. Cette tâche doit être effectuée par un administrateur de projet.

Avant de commencer

Remarque : Si vous prévoyez d'utiliser IBM MQ dans un projet dans lequel d'autres composants IBM Cloud Pak for Integration sont déjà installés, il n'est pas nécessaire de suivre les instructions ci-dessous.

Connectez-vous à votre cluster avec **`cloudctl login`** (pour IBM Cloud Pak for Integration) ou **`oc login`**.

Pourquoi et quand exécuter cette tâche

Les images de IBM MQ Operator sont extraites depuis un registre de conteneurs qui effectue un contrôle des autorisations de licence. Ce contrôle requiert une clé d'autorisation qui est stockée dans un secret d'extraction `docker-registry`. Si vous ne possédez pas encore de clé d'autorisation, suivez les instructions ci-après pour en obtenir une et créer un secret d'extraction.

Procédure

1. Obtenez la clé d'autorisation qui est affectée à votre ID.

a) Connectez-vous à Mon IBM - Bibliothèque des logiciels de conteneur avec l'ID et le mot de passe IBM associés au logiciel autorisé.

b) Dans la section **Clé d'autorisation**, cliquez sur **Copier la clé** pour copier la clé d'autorisation dans le presse-papiers.

2. Créez un secret contenant votre clé d'autorisation dans le projet dans lequel vous voulez déployer votre gestionnaire de files d'attente.

Exécutez la commande suivante, où *<entitlement-key>* est la clé extraite à l'étape 1, et *<user-email>* est l'ID IBM associé au logiciel autorisé.

```
oc create secret docker-registry ibm-entitlement-key \
--docker-server=cp.icr.io \
--docker-username=cp \
--docker-password=<entitlement-key> \
--docker-email=<user-email>
```

Que faire ensuite

«Déploiement d'un gestionnaire de files d'attente à l'aide de l'interface CLI Red Hat OpenShift», à la page 87

OpenShift CP4I Déploiement d'un gestionnaire de files d'attente sur un cluster Red Hat OpenShift Container Platform

Utilisez la ressource personnalisée QueueManager pour déployer un gestionnaire de files d'attente sur un cluster Red Hat OpenShift Container Platform.

Procédure

- CP4I
  «Déploiement d'un gestionnaire de files d'attente à l'aide d'IBM Cloud Pak for Integration Platform Navigator», à la page 85.
- OpenShift
  «Déploiement d'un gestionnaire de files d'attente à l'aide de la console Web Red Hat OpenShift», à la page 86.
- OpenShift
  «Déploiement d'un gestionnaire de files d'attente à l'aide de l'interface CLI Red Hat OpenShift», à la page 87.

Tâches associées

«Exemples de configuration d'un gestionnaire de files d'attente», à la page 89
Un gestionnaire de files d'attente peut être configuré en ajustant le contenu de la ressource personnalisée QueueManager.

CP4I *Déploiement d'un gestionnaire de files d'attente à l'aide d'IBM Cloud Pak for Integration Platform Navigator*

Utilisez la ressource personnalisée QueueManager pour déployer un gestionnaire de files d'attente dans un cluster Red Hat OpenShift Container Platform à l'aide d'IBM Cloud Pak for Integration Platform Navigator. Cette tâche doit être effectuée par un administrateur de projet.

Avant de commencer

Dans un navigateur, lancez IBM Cloud Pak for Integration Platform Navigator.

Si c'est la première fois qu'un gestionnaire de files d'attente est déployé dans ce projet Red Hat OpenShift, suivez les étapes présentées dans la rubrique «Préparation de votre projet Red Hat OpenShift pour IBM MQ», à la page 83.

Procédure

1. Déployez un gestionnaire de files d'attente.

   L'exemple ci-après déploie un gestionnaire de files d'attente "à démarrage rapide" qui utilise un stockage éphémère (non persistant) et désactive la sécurité MQ. Les messages ne seront pas conservés suite aux redémarrages du gestionnaire de files d'attente. Vous pouvez ajuster la configuration afin de changer de nombreux paramètres du gestionnaire de files d'attente.

a) Dans le IBM Cloud Pak for Integration Platform Navigator, cliquez sur **Administration** puis sur **Integration Runtimes**. Dans les anciennes versions de IBM Cloud Pak for Integration Platform Navigator, cliquez sur **Runtime and instances**.

b) Cliquez sur **Create instance**.

c) Sélectionnez **Messaging**, puis cliquez sur **Next**. Dans les anciennes versions de IBM Cloud Pak for Integration Platform Navigator, cliquez sur **Queue Manager**, puis sur **Next**.

Le formulaire de création d'instance d'une ressource `QueueManager` s'affiche.

Remarque : Vous pouvez aussi cliquer sur **Code** pour afficher ou changer le fichier YAML de configuration de la ressource `QueueManager`.

d) Dans la section **Details**, vérifiez ou mettez à jour la zone **Name** et spécifiez dans la zone **Namespace** l'espace de nom dans lequel créer l'instance de gestionnaire de files d'attente.

e) Si vous acceptez le contrat de licence d'IBM Cloud Pak for Integration, associez **License acceptance** à **On**.

Vous devez accepter la licence pour pouvoir déployer un gestionnaire de files d'attente.

f) Dans la section **Queue Manager**, cochez ou mettez à jour le **nom** du gestionnaire de files d'attente sous-jacent. Dans les anciennes versions de IBM Cloud Pak for Integration Platform Navigator, utilisez la section **Queue Manager Config**.

Par défaut, le nom du gestionnaire de files d'attente utilisé par les applications client IBM MQ est identique au nom de la ressource `QueueManager`, sans les éventuels caractères non valides (tels que les traits d'union), qui sont retirés.

g) Cliquez sur **Créer**.

La liste des gestionnaires de files d'attente qui se trouvent dans le projet (espace de nom) en cours est affichée. La nouvelle ressource `QueueManager` doit avoir le statut `Pending`.

2. Vérifiez que le gestionnaire de files d'attente est en cours d'exécution.

La création est terminée lorsque le statut de la ressource `QueueManager` est `Running`.

Tâches associées

«Configuration d'une route pour la connexion à un gestionnaire de files d'attente depuis l'extérieur d'un cluster Red Hat OpenShift», à la page 113
Vous avez besoin d'une route Red Hat OpenShift pour connecter une application à un gestionnaire de files d'attente IBM MQ depuis l'extérieur d'un cluster Red Hat OpenShift . Vous devez activer TLS sur votre gestionnaire de files d'attente et votre application client IBM MQ , car SNI est disponible uniquement dans le protocole TLS lorsqu'un protocole TLS 1.2 ou supérieur est utilisé. Red Hat OpenShift Container Platform Router utilise l'indication de nom de serveur pour acheminer les demandes vers le gestionnaire de files d'attente IBM MQ.

«Connexion à IBM MQ Console déployée dans un cluster Red Hat OpenShift», à la page 120
Connexion à la console IBM MQ Console d'un gestionnaire de files d'attente qui a été déployé dans un cluster Red Hat OpenShift Container Platform.

OpenShift CP4I *Déploiement d'un gestionnaire de files d'attente à l'aide de la console Web Red Hat OpenShift*

Utilisez la ressource personnalisée QueueManager pour déployer un gestionnaire de files d'attente dans un cluster Red Hat OpenShift Container Platform à l'aide de la console Web Red Hat OpenShift. Cette tâche doit être effectuée par un administrateur de projet.

Avant de commencer

Connectez-vous à votre console Web de cluster Red Hat OpenShift. Vous devez sélectionner un projet (espace de nom) existant ou en créez un.

Si c'est la première fois qu'un gestionnaire de files d'attente est déployé dans ce projet Red Hat OpenShift, suivez les étapes présentées dans la rubrique «Préparation de votre projet Red Hat OpenShift pour IBM MQ», à la page 83.

Procédure

1. Déployez un gestionnaire de files d'attente.

   L'exemple ci-après déploie un gestionnaire de files d'attente "à démarrage rapide" qui utilise un stockage éphémère (non persistant) et désactive la sécurité MQ. Les messages ne seront pas conservés suite aux redémarrages du gestionnaire de files d'attente. Vous pouvez ajuster la configuration afin de changer de nombreux paramètres du gestionnaire de files d'attente.

   a) Dans la console Web Red Hat OpenShift, dans le panneau de navigation, cliquez sur **Opérateurs** > **Opérateurs installés**

   b) Cliquez sur **IBM MQ**.

   c) cliquez sur l'onglet **Queue Manager**.

   d) Cliquez sur le bouton **Create QueueManager**.

      Un éditeur YAML contenant un exemple de fichier YAML pour une ressource `QueueManager` s'ouvre.

      Remarque : Vous pouvez aussi cliquer sur **Edit Form** pour afficher ou changer la configuration de `QueueManager`.

   e) Si vous acceptez le contrat de licence, associez **License acceptance** à **On**.

      IBM MQ est disponible avec plusieurs licences. Pour plus d'informations sur les licences valides, voir «Référence relative à l'octroi de licence pour mq.ibm.com/v1beta1», à la page 131. Vous devez accepter la licence pour pouvoir déployer un gestionnaire de files d'attente.

   f) Cliquez sur **Créer**.

      La liste des gestionnaires de files d'attente qui se trouvent dans le projet (espace de nom) en cours est affichée. La nouvelle ressource `QueueManager` doit être à l'état `Pending`.

2. Vérifiez que le gestionnaire de files d'attente est en cours d'exécution.

   La création est terminée lorsque le statut de la ressource `QueueManager` est `Running`.

Tâches associées

«Configuration d'une route pour la connexion à un gestionnaire de files d'attente depuis l'extérieur d'un cluster Red Hat OpenShift», à la page 113
Vous avez besoin d'une route Red Hat OpenShift pour connecter une application à un gestionnaire de files d'attente IBM MQ depuis l'extérieur d'un cluster Red Hat OpenShift . Vous devez activer TLS sur votre gestionnaire de files d'attente et votre application client IBM MQ , car SNI est disponible uniquement dans le protocole TLS lorsqu'un protocole TLS 1.2 ou supérieur est utilisé. Red Hat OpenShift Container Platform Router utilise l'indication de nom de serveur pour acheminer les demandes vers le gestionnaire de files d'attente IBM MQ.

«Connexion à IBM MQ Console déployée dans un cluster Red Hat OpenShift», à la page 120
Connexion à la console IBM MQ Console d'un gestionnaire de files d'attente qui a été déployé dans un cluster Red Hat OpenShift Container Platform.

OpenShift CP4I *Déploiement d'un gestionnaire de files d'attente à l'aide de l'interface CLI Red Hat OpenShift*

Utilisez la ressource personnalisée QueueManager pour déployer un gestionnaire de files d'attente dans un cluster Red Hat OpenShift Container Platform à l'aide de l'interface de ligne de commande (CLI). Cette tâche doit être effectuée par un administrateur de projet.

Avant de commencer

Vous devez installer l'interface de ligne de commande Red Hat OpenShift Container Platform.

Connectez-vous à votre cluster avec **`cloudctl login`** (pour IBM Cloud Pak for Integration) ou **`oc login`**.

Si c'est la première fois qu'un gestionnaire de files d'attente est déployé dans ce projet Red Hat OpenShift, suivez les étapes présentées dans la rubrique «Préparation de votre projet Red Hat OpenShift pour IBM MQ», à la page 83.

Procédure

1. Déployez un gestionnaire de files d'attente.

   L'exemple ci-après déploie un gestionnaire de files d'attente "à démarrage rapide" qui utilise un stockage éphémère (non persistant) et désactive la sécurité MQ. Les messages ne seront pas conservés suite aux redémarrages du gestionnaire de files d'attente. Vous pouvez ajuster le contenu du fichier YAML afin de changer de nombreux paramètres du gestionnaire de files d'attente.

   a) Création d'un fichier `QueueManager` YAML

      Par exemple, pour installer un gestionnaire de files d'attente de base dans IBM Cloud Pak for Integration, créez le fichier "mq-quickstart.yaml" dont le contenu est le suivant :

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: quickstart-cp4i
spec:
  version: 9.2.5.0-r3
  license:
    accept: false
    license: L-RJON-C7QG3S
    use: NonProduction
  web:
    enabled: true
  queueManager:
    name: "QUICKSTART"
    storage:
      queueManager:
        type: ephemeral
  template:
    pod:
      containers:
       - name: qmgr
         env:
         - name: MQSNOAUT
           value: "yes"
```

      Important : Si vous acceptez le contrat de licence IBM Cloud Pak for Integration, modifiez `accept: false` par `accept: true`. Voir «Référence relative à l'octroi de licence pour mq.ibm.com/v1beta1», à la page 131 pour des détails sur la licence.

      Cet exemple inclut également un serveur Web déployé avec le gestionnaire de files d'attente, où la console Web est activée pour la connexion unique avec IBM Cloud Pak Identity and Access Manager.

      Pour installer un gestionnaire de files d'attente de base indépendamment d'IBM Cloud Pak for Integration, créez le fichier "mq-quickstart.yaml" dont le contenu est le suivant :

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: quickstart
spec:
  version: 9.2.5.0-r3
  license:
    accept: false
    license: L-APIG-BZDDDY
  web:
    enabled: true
  queueManager:
    name: "QUICKSTART"
    storage:
      queueManager:
        type: ephemeral
  template:
    pod:
      containers:
       - name: qmgr
```

```
          env:
          - name: MQSNOAUT
            value: "yes"
```

Important : si vous acceptez le contrat de licence MQ, modifiez `accept: false` par `accept: true`. Voir «Référence relative à l'octroi de licence pour mq.ibm.com/v1beta1», à la page 131 pour des détails sur la licence.

b) Création de l'objet `QueueManager`

```
oc apply -f mq-quickstart.yaml
```

2. Vérifiez que le gestionnaire de files d'attente est en cours d'exécution.

   Vous pouvez valider le déploiement en exécutant

```
oc describe queuemanager <QueueManagerResourceName>
```

   , puis en vérifiant le statut.

   Par exemple, exécutez

```
oc describe queuemanager quickstart
```

   , et vérifiez que la zone `status.Phase` indique `Running`

Tâches associées

«Configuration d'une route pour la connexion à un gestionnaire de files d'attente depuis l'extérieur d'un cluster Red Hat OpenShift», à la page 113
Vous avez besoin d'une route Red Hat OpenShift pour connecter une application à un gestionnaire de files d'attente IBM MQ depuis l'extérieur d'un cluster Red Hat OpenShift . Vous devez activer TLS sur votre gestionnaire de files d'attente et votre application client IBM MQ , car SNI est disponible uniquement dans le protocole TLS lorsqu'un protocole TLS 1.2 ou supérieur est utilisé. Red Hat OpenShift Container Platform Router utilise l'indication de nom de serveur pour acheminer les demandes vers le gestionnaire de files d'attente IBM MQ.

«Connexion à IBM MQ Console déployée dans un cluster Red Hat OpenShift», à la page 120
Connexion à la console IBM MQ Console d'un gestionnaire de files d'attente qui a été déployé dans un cluster Red Hat OpenShift Container Platform.

OpenShift CP4I Exemples de configuration d'un gestionnaire de files d'attente

Un gestionnaire de files d'attente peut être configuré en ajustant le contenu de la ressource personnalisée QueueManager.

Pourquoi et quand exécuter cette tâche

Utilisez les exemples suivants pour vous aider à configurer un gestionnaire de files d'attente à l'aide du fichier YAML QueueManager.

Procédure

- «Exemple : fourniture de fichiers MQSC et INI», à la page 89
- «Exemple : configuration de TLS», à la page 91

OpenShift CP4I *Exemple : fourniture de fichiers MQSC et INI*

Cet exemple crée une mappe de configuration Kubernetes contenant deux fichiers MQSC et un fichier INI. Un gestionnaire de files d'attente qui traite ces fichiers MQSC et INI est alors déployé.

Pourquoi et quand exécuter cette tâche

Les fichiers MQSC et INI peuvent être fournis lorsqu'un gestionnaire de files d'attente est déployé. Les données MQSC et INI sont définies dans un ou plusieurs Kubernetes ConfigMaps et Secrets. Ces éléments doivent être créés dans l'espace de nom (projet) où vous déployez le gestionnaire de files d'attente.

Remarque : Un secret Kubernetes doit être utilisé si le fichier MQSC ou INI contient des données sensibles.

La fourniture de MQSC et INI de cette manière requiert IBM MQ Operator 1.1 ou une version ultérieure.

Exemple

L'exemple suivant crée une mappe de configuration Kubernetes contenant deux fichiers MQSC et un fichier INI. Un gestionnaire de files d'attente qui traite ces fichiers MQSC et INI est alors déployé.

Exemple de mappe de configuration (ConfigMap) - appliquez le code YAML suivant dans votre cluster :

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: mqsc-ini-example
data:
  example1.mqsc: |
    DEFINE QLOCAL('DEV.QUEUE.1') REPLACE
    DEFINE QLOCAL('DEV.QUEUE.2') REPLACE
  example2.mqsc: |
    DEFINE QLOCAL('DEV.DEAD.LETTER.QUEUE') REPLACE
  example.ini: |
    Channels:
       MQIBindType=FASTPATH
```

Exemple de gestionnaire de files d'attente (QueueManager) - déployez le gestionnaire de files d'attente avec la configuration suivante, à l'aide de la ligne de commande ou de IBM Cloud Pak for Integration Platform Navigator :

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: mqsc-ini-cp4i
spec:
  version: 9.2.5.0-r3
  license:
    accept: false
    license: L-RJON-C7QG3S
    use: NonProduction
  web:
    enabled: true
  queueManager:
    name: "MQSCINI"
    mqsc:
    - configMap:
        name: mqsc-ini-example
        items:
        - example1.mqsc
        - example2.mqsc
    ini:
    - configMap:
        name: mqsc-ini-example
        items:
        - example.ini
    storage:
      queueManager:
        type: ephemeral
```

Important : Si vous acceptez le contrat de licence IBM Cloud Pak for Integration, modifiez `accept: false` par `accept: true`. Pour plus d'informations sur la licence, voir Référence relative à l'octroi de licence pour mq.ibm.com/v1beta1.

Informations supplémentaires :

- Un gestionnaire de files d'attente peut être configuré pour utiliser un seul élément ConfigMap ou Secret Kubernetes (comme illustré dans cet exemple) ou plusieurs éléments ConfigMap et Secret Kubernetes.
- Vous pouvez choisir d'utiliser toutes les données MQSC et INI à partir d'un élément ConfigMap ou Secret Kubernetes (comme indiqué dans cet exemple) ou de configurer chaque gestionnaire de files d'attente pour qu'il n'utilise qu'un sous-ensemble des fichiers disponibles.
- Les fichiers MQSC et INI sont traités par ordre alphabétique en fonction de leur clé. Ainsi, `example1.mqsc` sera toujours traité avant `example2.mqsc`, quel que soit l'ordre dans lequel ils apparaissent dans la configuration du gestionnaire de files d'attente.
- Si plusieurs fichiers MQSC ou INI possèdent la même clé, entre plusieurs éléments ConfigMap ou Secret Kubernetes, cet ensemble de fichiers est traité en fonction de l'ordre dans lequel les fichiers sont définis dans la configuration du gestionnaire de files d'attente.

OpenShift CP4I Linux ***Exemple : configuration de TLS***

Cet exemple déploie un gestionnaire de files d'attente dans Red Hat OpenShift Container Platform à l'aide de IBM MQ Operator. La communication TLS unidirectionnelle est configurée entre un client exemple et le gestionnaire de files d'attente. L'exemple illustre la réussite de la configuration en plaçant et en obtenant des messages.

Avant de commencer

Pour mettre en oeuvre cet exemple, vous devez d'abord avoir rempli les conditions suivantes :

- Installez IBM MQ clientet ajoutez `samp/bin` et `bin` à votre *CHEMIN*. Vous avez besoin des applications **`runmqakm`**, **`amqsputc`** et **`amqsgetc`**, qui peuvent être installées en tant que composant de IBM MQ client comme suit :
  - Windows Linux Pour Windows et Linux : installez le client redistribuable IBM MQ pour votre système d'exploitation à partir de https://ibm.biz/mq92redistclients
  - macOS Pour Mac: téléchargez et configurez IBM MQ MacOS Toolkit: https://developer.ibm.com/tutorials/mq-macos-dev/
- Installez l'outil OpenSSL correspondant à votre système d'exploitation.
- Créez un projet / espace de nom Red Hat OpenShift Container Platform (OCP) pour cet exemple.
- Sur la ligne de commande, connectez-vous au cluster OCP et passez à l'espace de nom ci-dessus.
- Vérifiez que IBM MQ Operator est installé et disponible dans l'espace de nom ci-dessus.

Pourquoi et quand exécuter cette tâche

Cet exemple fournit une ressource personnalisée YAML définissant un gestionnaire de files d'attente à déployer dans Red Hat OpenShift Container Platform. Il détaille également les étapes supplémentaires requises pour déployer le gestionnaire de files d'attente avec TLS activé. Une fois l'exécution terminée, la mise en place et l'extraction des messages valident le gestionnaire de files d'attente configuré avec TLS.

Création d'une clé privée et de certificats TLS pour le serveur IBM MQ

Les exemples de code suivants montrent comment créer un certificat autosigné pour le gestionnaire de files d'attente et comment ajouter le certificat à une base de données de clés pour agir en tant que fichier de clés certifiées pour le client. Si vous disposez déjà d'une clé privée et d'un certificat, vous pouvez les utiliser à la place.

Notez que les certificats autosignés ne doivent être utilisés qu'à des fins de développement.

Création d'une clé privée autosignée et d'un certificat public dans le répertoire en cours

Exécutez ensuite la commande suivante :

```
openssl req -newkey rsa:2048 -nodes -keyout tls.key -subj "/CN=localhost" -x509 -days 3650
-out tls.crt
```

Ajout de la clé publique du serveur à une base de données de clés client

La base de données de clés est utilisée comme fichier de clés certifiées pour l'application client.

Créez la base de données de clés client :

```
runmqakm -keydb -create -db clientkey.kdb -pw password -type cms -stash
```

Ajoutez la clé publique précédemment générée à la base de données de clés client :

```
runmqakm -cert -add -db clientkey.kdb -label mqservercert -file tls.crt -format ascii
-stashed
```

Configuration des certificats TLS pour le déploiement du gestionnaire de files d'attente

Pour que votre gestionnaire de files d'attente puisse référencer et appliquer la clé et le certificat, créez un secret TLS Kubernetes qui fait référence aux fichiers créés ci-dessus. Ce faisant, vérifiez que vous êtes dans l'espace de nom que vous avez créé avant le début de cette tâche.

```
oc create secret tls example-tls-secret --key="tls.key" --cert="tls.crt"
```

Créez une mappe de configuration contenant des commandes MQSC

Créez une mappe de configuration Kubernetes contenant les commandes MQSC pour créer une file d'attente et un canal SVRCONN, et pour ajouter un enregistrement d'authentification de canal qui autorise l'accès au canal en bloquant uniquement les utilisateurs appelés *nobody*.

Notez que cette approche ne doit être utilisée qu'à des fins de développement.

Vérifiez que vous êtes dans l'espace de nom que vous avez créé précédemment (voir Avant de commencer), puis entrez le code YAML suivant dans l'interface utilisateur OCP ou utilisez la ligne de commande.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: example-tls-configmap
data:
  tls.mqsc: |
    DEFINE QLOCAL('EXAMPLE.QUEUE') REPLACE
    DEFINE CHANNEL(SECUREQMCHL) CHLTYPE(SVRCONN) TRPTYPE(TCP) SSLCAUTH(OPTIONAL)
SSLCIPH('ANY_TLS12_OR_HIGHER')
    SET CHLAUTH(SECUREQMCHL) TYPE(BLOCKUSER) USERLIST('nobody') ACTION(ADD)
```

Créez la route OCP requise

Vérifiez que vous êtes dans l'espace de nom que vous avez créé avant le début de cette tâche, puis entrez le code YAML dans l'interface utilisateur OCP ou utilisez la ligne de commande.

```
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  name: example-tls-route
spec:
  host: secureqmchl.chl.mq.ibm.com
  to:
    kind: Service
    name: secureqm-ibm-mq
  port:
    targetPort: 1414
  tls:
    termination: passthrough
```

Notez que le Red Hat OpenShift Container Platform Router utilise SNI pour acheminer les demandes vers le gestionnaire de files d'attente IBM MQ. Si vous modifiez le nom de canal indiqué dans le MQSC dans la mappe de configuration créée précédemment, vous devez également modifier la zone hôte ici et dans le fichier CCDT créé ultérieurement. Pour plus d'informations, voir «Configuration d'une route pour la connexion à un gestionnaire de files d'attente depuis l'extérieur d'un cluster Red Hat OpenShift», à la page 113.

Déployez le gestionnaire de files d'attente

Important : dans cet exemple, nous utilisons la variable *MQSNOAUT* pour désactiver l'autorisation sur le gestionnaire de files d'attente, ce qui nous permet de nous concentrer sur les étapes requises pour connecter un client à l'aide de TLS. Cette action n'est pas recommandée dans un déploiement de production d'IBM MQ, car il en résulte que toutes les applications qui se connectent disposent des pleins pouvoirs administratifs, sans mécanisme permettant de réduire les droits d'accès pour chaque application..

Créez un gestionnaire de files d'attente à l'aide de la ressource personnalisée YAML suivante. Notez qu'elle fait référence à la mappe de configuration et au secret créés précédemment, ainsi qu'à la variable *MQSNOAUT*.

Vérifiez que vous êtes dans l'espace de nom que vous avez créé avant de commencer cette tâche, puis entrez le code YAML suivant dans l'interface utilisateur OCP, à l'aide de la ligne de commande ou de IBM Cloud Pak for Integration Platform Navigator. Vérifiez que la licence est correcte et acceptez la licence en modifiant `false` par `true`.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: secureqm
spec:
  license:
    accept: false
    license: L-RJON-C7QG3S
    use: Production
  queueManager:
    name: SECUREQM
    mqsc:
    - configMap:
        name: example-tls-configmap
        items:
        - tls.mqsc
    storage:
      queueManager:
        type: ephemeral
  template:
    pod:
      containers:
        - env:
            - name: MQSNOAUT
              value: 'yes'
          name: qmgr
  version: 9.2.5.0-r3
  web:
    enabled: true
  pki:
    keys:
      - name: example
        secret:
          secretName: example-tls-secret
          items:
          - tls.key
          - tls.crt
```

Confirmez que le gestionnaire de files d'attente est en cours d'exécution

Le gestionnaire de files d'attente est en cours de déploiement. Confirmez qu'il se trouve dans l'état `Running` avant de continuer. Exemple :

```
oc get qmgr secureqm
```

Testez la connexion au gestionnaire de files d'attente

Pour confirmer que le gestionnaire de files d'attente est configuré pour une communication TLS unidirectionnelle, utilisez les exemples d'applications **amqsputc** et **amqsgetc** :

Localisez le nom d'hôte du gestionnaire de files d'attente

Utilisez la commande suivante pour rechercher le nom d'hôte complet du gestionnaire de files d'attente pour la route `secureqm-ibm-mq-qm` :

```
oc get routes secureqm-ibm-mq-qm
```

Spécifiez les détails du gestionnaire de files d'attente

Créez un fichier `CCDT.JSON` qui spécifie les détails du gestionnaire de files d'attente. Remplacez la valeur de l'hôte par le nom d'hôte de l'étape précédente.

```
{
    "channel":
    [
        {
            "name": "SECUREQMCHL",
            "clientConnection":
            {
                "connection":
                [
                    {
                        "host": "<hostname from previous step>",
                        "port": 443
                    }
                ],
                "queueManager": "SECUREQM"
            },
            "transmissionSecurity":
            {
              "cipherSpecification": "ECDHE_RSA_AES_128_CBC_SHA256"
            },
            "type": "clientConnection"
        }
   ]
}
```

Exportez les variables d'environnement

Exportez les variables d'environnement suivantes, de la manière appropriée pour votre système d'exploitation. Ces variables seront lues par **amqsputc** et **amqsgetc**.

Mettez à jour le chemin d'accès aux fichiers sur votre système :

```
export MQCCDTURL='<full path to file>/CCDT.JSON'
export MQSSLKEYR='<full path to file>/clientkey'
```

Insérez des messages dans la file d'attente

Exécutez ensuite la commande suivante :

```
amqsputc EXAMPLE.QUEUE SECUREQM
```

Si la connexion au gestionnaire de files d'attente aboutit, la réponse suivante apparaît :

```
target queue is EXAMPLE.QUEUE
```

Placez plusieurs messages dans la file d'attente en entrant un texte, puis en appuyant sur **Entrée** à chaque fois.

Pour terminer, appuyez deux fois sur **Entrée**.

Extrayez les messages de la file d'attente

Exécutez ensuite la commande suivante :

```
amqsgetc EXAMPLE.QUEUE SECUREQM
```

Les messages que vous avez ajoutés à l'étape précédente ont été utilisés et sont renvoyés.

Après quelques secondes, la commande prend fin.

Félicitations, vous avez déployé avec succès un gestionnaire de files d'attente avec TLS activé et montré que vous pouvez insérer et extraire des messages en toute sécurité sur le gestionnaire de files d'attente à partir d'un client.

OpenShift CP4I *Exemple : Personnalisation des annotations de service de licence*

Le IBM MQ Operator ajoute automatiquement des annotations IBM License Service aux ressources déployées. Ils sont surveillés par IBM License Service, et des rapports correspondant à l'autorisation requise sont générés.

Pourquoi et quand exécuter cette tâche

Les annotations ajoutées par IBM MQ Operator sont celles attendues dans des situations standard et sont basées sur les valeurs de licence sélectionnées lors du déploiement d'un gestionnaire de files d'attente.

Exemple

Si **License** est défini sur `L-RJON-BZFQU2 (IBM Cloud Pak for Integration 2021.2.1)`et que **Use** est défini sur `Nonproduction`, les annotations suivantes sont appliquées :

- cloudpakId: c8b82d189e7545f0892db9ef2731b90d
- cloudpakName: IBM Cloud Pak for Integration
- productChargedContainers : qmgr
- productCloudpakRatio : '4:1'
- productID: 21dfe9a0f00f444f888756d835334909
- productName : IBM MQ Advanced for Non-Production
- ProductMetric : VIRTUAL_PROCESSOR_CORE
- productVersion: 9.2.3.0

Dans IBM Cloud Pak for Integration, les déploiements de IBM App Connect Enterprise incluent une autorisation restreinte pour IBM MQ. Dans ces situations, ces annotations doivent être remplacées pour garantir que IBM License Service capture l'utilisation correcte. Pour ce faire, utilisez l'approche décrite dans «Ajout d'annotations et d'étiquettes personnalisées aux ressources du gestionnaire de files d'attente», à la page 118.

Par exemple, si IBM MQ est déployé sous IBM App Connect Enterprise, utilisez l'approche illustrée dans le fragment de code suivant :

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: mq4ace
  namespace: cp4i
spec:
  annotations:
    productMetric: FREE
```

Il existe deux autres raisons pour lesquelles des annotations de licence peuvent être modifiées :

1. IBM MQ Advanced est inclus dans l'autorisation d'un autre produit IBM.
   - Dans ce cas, utilisez l'approche précédemment décrite pour IBM App Connect Enterprise.
2. IBM MQ est déployé sous une licence IBM Cloud Pak for Integration.
   - Si vous disposez d'une licence IBM Cloud Pak for Integration, vous pouvez décider de déployer un gestionnaire de files d'attente sous le rapport IBM MQ ou IBM MQ Advanced. Si vous effectuez un déploiement avec un rapport IBM MQ, vous devez vous assurer que vous n'utilisez pas de fonctions avancées telles que la haute disponibilité native ou Advanced Message Security.
   - Dans ce cas, utilisez les annotations suivantes pour l'utilisation en production :

     ```
     apiVersion: mq.ibm.com/v1beta1
     kind: QueueManager
     metadata:
       name: mq4ace
       namespace: cp4i
     ```

```
spec:
  annotations:
    productID: c661609261d5471fb4ff8970a36bccea
    productCloudpakRatio: '4:1'
    productName: IBM MQ for Production
    productMetric: VIRTUAL_PROCESSOR_CORE
```

- Utilisez les annotations suivantes pour une utilisation autre que la production :

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: mq4ace
  namespace: cp4i
spec:
  annotations:
    productID: 151bec68564a4a47a14e6fa99266deff
    productCloudpakRatio: '8:1'
    productName: IBM MQ for Non-Production
    productMetric: VIRTUAL_PROCESSOR_CORE
```

OpenShift CP4I Configuration de la haute disponibilité pour les gestionnaires de files d'attente à l'aide de IBM MQ Operator

Pourquoi et quand exécuter cette tâche

Procédure

- V 9.2.3
  «Native HA», à la page 96.
- V 9.2.3
  «Exemple : configuration d'un gestionnaire de files d'attente Native HA», à la page 98.
- «Exemple : configuration d'un gestionnaire de files d'attente multi-instance», à la page 107.

CP4I V 9.2.3 CD *Native HA*

Native HA est une solution de haute disponibilité native (intégrée) pour IBM MQ qui peut être utilisée avec le stockage par blocs sur cloud.

Une configuration Native HA fournit un gestionnaire de files d'attente hautement disponible dans lequel les données MQ récupérables (par exemple, les messages) sont répliquées sur plusieurs ensembles de stockage, ce qui empêche toute perte de données en cas d'incidents de stockage. Le gestionnaire de files d'attente est constitué de plusieurs instances actives, l'une étant l'instance principale et les autres étant prêtes à prendre rapidement le relais en cas d'échec, afin de maximiser l'accès au gestionnaire de files d'attente et à ses messages.

Une configuration Native HA configuration est constituée de trois pods Kubernetes, chacun contenant une instance du gestionnaire de files d'attente. Une instance correspond au gestionnaire de files d'attente actif, qui traite les messages et écrit dans son journal de reprise. A chaque écriture dans le journal de reprise, le gestionnaire de files d'attente actif envoie les données aux deux autres instances, appelées répliques. Chaque réplique écrit dans son propre journal de reprise, reconnaît les données, puis met à jour ses propres données de file d'attente à partir du journal de reprise répliqué. Si le pod qui exécute le gestionnaire de files d'attente actif échoue, l'une des répliques d'instance du gestionnaire de files d'attente devient actif et dispose des données à jour qu'il peut utiliser.

Le type de journal est appelé "journal répliqué". Un journal répliqué est essentiellement un journal linéaire, avec une gestion automatique des journaux et des images de support automatiques activées. Voir Types de journalisation. Vous utilisez les mêmes techniques de gestion du journal répliqué que celles utilisées pour la gestion d'un journal linéaire.

Un service Kubernetes est utilisé pour acheminer les connexions client TCP/IP à l'instance active en cours, identifiée comme étant le seul pod prêt pour le trafic réseau. Cela se produit sans qu'il soit nécessaire que l'application client ait conscience des différentes instances.

Trois pods sont utilisés pour réduire considérablement la possibilité d'une situation de split-brain. Dans un système haute disponibilité à deux pods, split-brain peut se produire lorsque la connectivité entre les deux pods est rompue. En l'absence de connectivité, les deux pods peuvent exécuter le gestionnaire de files d'attente en même temps et accumuler des données différentes. Lorsque la connexion est restaurée, il existe alors deux versions différentes des données (un "split-brain') et une intervention manuelle est requise pour déterminer les données à conserver et celles à supprimer.

Native HA utilise un système à trois pod avec quorum pour éviter une situation de split-brain. Les pods qui peuvent communiquer avec au moins l'un des autres pods constituent le quorum. Un gestionnaire de files d'attente ne peut devenir l'instance active que sur un pod qui dispose du quorum. Le gestionnaire de files d'attente ne pouvant pas devenir actif sur un pod non connecté à au moins un autre pod, il ne peut donc jamais exister deux instances actives en même temps :

- En cas de défaillance d'un seul pod, le gestionnaire de files d'attente sur l'un des deux autres pods peut prendre le relais. En cas de défaillance de deux pods, le gestionnaire de files d'attente ne peut pas devenir l'instance active sur le pod restant car ce pod ne dispose pas du quorum (le pod restant ne peut pas savoir si les deux autres pods ont échoué ou s'ils sont toujours actifs et qu'il a perdu la connectivité).
- Si un seul pod perd la connectivité, le gestionnaire de files d'attente ne peut pas devenir actif sur ce pod car le pod ne dispose pas du quorum. Le gestionnaire de files d'attente sur l'un des deux pods restants, qui dispose du quorum, peut prendre le relais. Si tous les pods perdent la connectivité, le gestionnaire de files d'attente ne peut pas devenir actif sur l'un des pods car aucun ne dispose du quorum.

Si un pod actif échoue, puis est restauré, il peut rejoindre le groupe avec le rôle de réplique.

La figure ci-après illustre un déploiement type avec trois instances d'un gestionnaire de files d'attente déployé dans trois conteneurs.

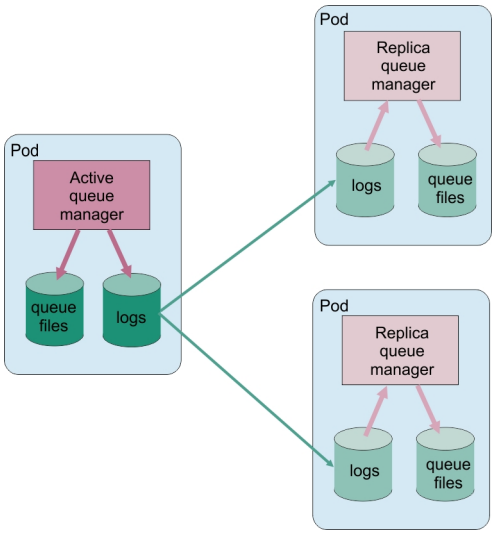


Figure 1. Exemple de configuration Native HA

CP4I V9.2.3 CD *Configuration de Native HA à l'aide de IBM MQ Operator*

Native HA est configurée à l'aide de l'API `QueueManager`, et des options avancées sont disponibles à l'aide d'un fichier INI.

Native HA est configurée à l'aide de `.spec.queueManager.availability` de l'API `QueueManager`, par exemple :

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
```

```
metadata:
name: nativeha-example
spec:
  license:
    accept: false
    license: L-RJON-C7QG3S
    use: Production
  queueManager:
    availability:
      type: NativeHA
  version: 9.2.5.0-r3
```

La zone `.spec.queueManager.availability.type` doit être définie sur `NativeHA`.

Native HA est disponible dans IBM MQ 9.2.3 ou versions supérieures.

Sous `.spec.queueManager.availability`, vous pouvez également configurer un secret TLS et des chiffrements à utiliser entre les instances de gestionnaire de files d'attente lors de la réplication. Cela est fortement recommandé et un guide pas à pas est disponible dans la rubrique «Exemple : configuration d'un gestionnaire de files d'attente Native HA», à la page 98.

Référence associée

«Exemple : configuration d'un gestionnaire de files d'attente Native HA», à la page 98
Cet exemple montre comment déployer un gestionnaire de files d'attente à l'aide de la fonctionnalité de haute disponibilité native dans OCP (Red Hat OpenShift Container Platform) à l'aide d'IBM MQ Operator.

Exemple : configuration d'un gestionnaire de files d'attente Native HA

Cet exemple montre comment déployer un gestionnaire de files d'attente à l'aide de la fonctionnalité de haute disponibilité native dans OCP (Red Hat OpenShift Container Platform) à l'aide d'IBM MQ Operator.

Avant de commencer

Pour mettre en oeuvre cet exemple, vous devez d'abord avoir rempli les conditions suivantes :

- Installez IBM MQ client et ajoutez les répertoires `samp/bin` et `bin` installés à votre *CHEMIN*. Le client fournit les applications **`runmqakm`**, **`amqsputc`** et **`amqsgetc`** requises par cet exemple. Installez le IBM MQ client comme suit :
  - 

Pour Windows et Linux : installez le client redistribuable IBM MQ pour votre système d'exploitation à partir de https://ibm.biz/mq92redistclients
  - 

Pour Mac : téléchargez et configurez IBM MQ MacOS Toolkit. Voir https://ibm.biz/mqdevmacclient.
- Installez l'outil OpenSSL correspondant à votre système d'exploitation. Vous en avez besoin pour générer un certificat autosigné pour le gestionnaire de files d'attente, si vous ne disposez pas déjà d'une clé privée et d'un certificat.
- Créez un projet / espace de nom Red Hat OpenShift Container Platform (OCP) pour cet exemple et suivez les étapes de la tâche «Préparation de votre projet Red Hat OpenShift pour IBM MQ», à la page 83
- Sur la ligne de commande, connectez-vous au cluster OCP, puis passez à l'espace de nom que vous venez de créer.
- Vérifiez qu'IBM MQ Operator est installé et disponible dans l'espace de nom.
- Configurez une classe de stockage par défaut dans OCP, à utiliser par votre gestionnaire de files d'attente. Si vous souhaitez suivre ce tutoriel sans définir de classe de stockage par défaut, reportez-vous à la rubrique Remarque 2 : utilisation d'une classe de stockage autre que celle par défaut.

A propos de cette tâche

Les gestionnaires de files d'attente Native HA impliquent un pod Kubernetes actif et deux répliques. Ils sont exécutés dans le cadre d'un objet StatefulSet Kubernetes avec exactement trois répliques et

un ensemble de volumes persistants Kubernetes. Pour plus d'informations sur les gestionnaires de files d'attente Native HA, reportez-vous à la rubrique «Haute disponibilité pour IBM MQ dans les conteneurs», à la page 16.

L'exemple fournit une ressource personnalisée YAML qui définit un gestionnaire de files d'attente Native HA qui utilise un stockage persistant et est configuré avec TLS. Une fois que vous avez déployé le gestionnaire de files d'attente dans OCP, simulez une défaillance du pod du gestionnaire de files d'attente actif. Vous pouvez constater que la reprise automatique a été déclenchée et vérifier qu'elle a abouti en envoyant et recevant des messages après l'incident.

Exemple

Création d'une clé privée et de certificats TLS pour le serveur MQ

Vous pouvez créer un certificat autosigné pour le gestionnaire de files d'attente et ajouter ce certificat à une base de données de clés pour agir en tant que fichier de clés certifiées du client. Si vous disposez déjà d'une clé privée et d'un certificat, vous pouvez les utiliser à la place. Notez que vous ne devez utiliser des certificats autosignés qu'à des fins de développement.

Pour créer une clé privée autosignée et un certificat public dans le répertoire de travail, exécutez la commande suivante :

```
openssl req -newkey rsa:2048 -nodes -keyout tls.key -subj "/CN=localhost" -x509 -days 3650
-out tls.crt
```

Création d'une clé privée et de certificats TLS à utiliser en interne par Native HA

Les trois pods d'un gestionnaire de files d'attente Native HA répliquent les données sur le réseau. Vous pouvez créer un certificat autosigné à utiliser lors de la réplication en interne. Notez que vous ne devez utiliser des certificats autosignés qu'à des fins de développement.

Pour créer une clé privée autosignée et un certificat public dans le répertoire de travail, exécutez la commande suivante :

```
openssl req -newkey rsa:2048 -nodes -keyout nativeha.key -subj "/CN=localhost" -x509 -days
3650 -out nativeha.crt
```

Ajout de la clé publique du gestionnaire de files d'attente à une base de données de clés client

Une base de données de clés client est utilisée comme fichier de clés certifiées pour l'application client.

Créez la base de données de clés client :

```
runmqakm -keydb -create -db clientkey.kdb -pw password -type cms -stash
```

Ajoutez la clé publique précédemment générée à la base de données de clés client :

```
runmqakm -cert -add -db clientkey.kdb -label mqservercert -file tls.crt -format ascii
-stashed
```

Création d'un secret contenant les certificats TLS pour le déploiement du gestionnaire de files d'attente

Pour que votre gestionnaire de files d'attente puisse référencer et appliquer la clé et le certificat, créez un secret TLS Kubernetes qui fait référence aux fichiers créés ci-dessus. Ce faisant, vérifiez que vous êtes dans l'espace de nom que vous avez créé avant le début de cette tâche.

```
oc create secret tls example-ha-secret --key="tls.key" --cert="tls.crt"
```

Création d'un secret contenant la clé et le certificat TLS Native HA internes

Pour que votre gestionnaire de files d'attente puisse référencer et appliquer la clé et le certificat, créez un secret TLS Kubernetes qui fait référence aux fichiers créés ci-dessus. Ce faisant, vérifiez que vous êtes dans l'espace de nom que vous avez créé avant le début de cette tâche.

```
oc create secret tls example-ha-secret-internal --key="nativeha.key" --cert="nativeha.crt"
```

Créez une mappe de configuration contenant des commandes MQSC

Créez une mappe de configuration Kubernetes contenant les commandes MQSC pour créer une file d'attente et un canal SVRCONN, et pour ajouter un enregistrement d'authentification de canal qui autorise l'accès au canal en bloquant uniquement les utilisateurs appelés *nobody*.

Notez que cette approche ne doit être utilisée qu'à des fins de développement.

Vérifiez que vous vous trouvez dans l'espace de nom que vous avez créé précédemment (voir «Avant de commencer», à la page 98), puis entrez le code YAML suivant dans l'interface utilisateur OCP ou utilisez la ligne de commande :

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: example-mi-configmap
data:
  tls.mqsc: |
    DEFINE QLOCAL('EXAMPLE.QUEUE') DEFPSIST(YES) REPLACE
    DEFINE CHANNEL(HAQMCHL) CHLTYPE(SVRCONN) TRPTYPE(TCP) SSLCAUTH(OPTIONAL)
SSLCIPH('ANY_TLS12_OR_HIGHER')
    SET CHLAUTH(HAQMCHL) TYPE(BLOCKUSER) USERLIST('nobody') ACTION(ADD)
```

Configuration du routage

Si vous utilisez un IBM MQ client ou un kit d'outils IBM MQ 9.2.1 (ou version ultérieure), vous pouvez configurer le routage vers le gestionnaire de files d'attente à l'aide d'un fichier de configuration de gestionnaire de files d'attente (fichier INI). Dans ce fichier, vous définissez la variable *OutboundSNI* pour un routage en fonction du nom d'hôte et non du nom de canal.

Créez un fichier dans le répertoire dans lequel vous exécutez des commandes, nommé `mqclient.ini`, contenant exactement le texte suivant :

```
SSL:
   OutboundSNI=HOSTNAME
```

Ne modifiez aucune valeur de ce fichier INI. Par exemple, la chaîne HOSTNAME ne doit pas être modifiée.

Pour plus d'informations, reportez-vous à la rubrique Strophe SSL du fichier de configuration client.

Si vous utilisez un IBM MQ client ou un kit d'outils antérieur à IBM MQ 9.2.1, vous devez créer un chemin OCP au lieu du fichier de configuration précédent. Suivez les étapes de la rubrique Remarque 1 : création d'un chemin.

Déployez le gestionnaire de files d'attente

Important : dans cet exemple, nous utilisons la variable *MQSNOAUT* pour désactiver l'autorisation sur le gestionnaire de files d'attente, ce qui nous permet de nous concentrer sur les étapes requises pour connecter un client à l'aide de TLS. Cette action n'est pas recommandée dans un déploiement de production d'IBM MQ, car il en résulte que toutes les applications qui se connectent disposent des pleins pouvoirs administratifs, sans mécanisme permettant de réduire les droits d'accès pour chaque application..

Copiez et mettez à jour le fichier YAML ci-après.

- Vérifiez que la licence appropriée est spécifiée. Voir Référence relative à l'octroi de licence pour mq.ibm.com/v1beta1. Dans IBM Cloud Pak for Integration 2021.1.1, la licence doit être la licence d'évaluation `L-RJON-BYRMYW`
- Acceptez la licence en remplaçant `false` par `true`.

Fichier YAML des ressources personnalisées du gestionnaire de files d'attente

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: nativeha-example
spec:
  license:
    accept: false
```

```
  license: L-RJON-C7QG3S
  use: Production
queueManager:
  name: HAEXAMPLE
  availability:
    type: NativeHA
    tls:
      secretName: example-ha-secret-internal
  mqsc:
  - configMap:
      name: example-mi-configmap
      items:
      - tls.mqsc
template:
  pod:
    containers:
      - env:
          - name: MQSNOAUT
            value: 'yes'
        name: qmgr
version: 9.2.5.0-r3
pki:
  keys:
    - name: example
      secret:
        secretName: example-ha-secret
        items:
        - tls.key
        - tls.crt
```

Pour vous assurer que vous vous trouvez dans l'espace de nom créé précédemment, déployez le fichier YAML mis à jour à l'aide de la console Web Red Hat OpenShift Container Platform, de la ligne de commande ou d'IBM Cloud Pak for Integration Platform Navigator.

Il y a un bref délai pendant que le système configure le gestionnaire de files d'attente Native HA, après quoi le gestionnaire de files d'attente doit être disponible pour utilisation.

Confirmation

Dans cette section, nous nous assurons que le gestionnaire de files d'attente se comporte comme prévu.

Confirmez que le gestionnaire de files d'attente est en cours d'exécution

Le gestionnaire de files d'attente est en cours de déploiement. Confirmez qu'il se trouve dans l'état `Running` avant de continuer. Exemple :

```
oc get qmgr nativeha-example
```

Testez la connexion au gestionnaire de files d'attente

Pour confirmer que le gestionnaire de files d'attente est configuré pour une communication TLS unidirectionnelle, utilisez les exemples d'applications **amqsputc** et **amqsgetc** :

Localisez le nom d'hôte du gestionnaire de files d'attente

Pour rechercher le nom d'hôte du gestionnaire de files d'attente pour la route `nativeha-example-ibm-mq-qm`, exécutez la commande suivante. Le nom d'hôte est renvoyé dans la zone `HOST`.

```
oc get routes nativeha-example-ibm-mq-qm
```

Spécifiez les détails du gestionnaire de files d'attente

Créez un fichier `CCDT.JSON` qui spécifie les détails du gestionnaire de files d'attente. Remplacez la valeur de l'hôte par le nom d'hôte renvoyé par l'étape précédente.

```
{
    "channel":
    [
        {
            "name": "HAQMCHL",
            "clientConnection":
            {
                "connection":
                [
                    {
```

```
                        "host": "<host from previous step>",
                        "port": 443
                    }
                ],
                "queueManager": "HAEXAMPLE"
            },
            "transmissionSecurity":
            {
              "cipherSpecification": "ECDHE_RSA_AES_128_CBC_SHA256"
            },
            "type": "clientConnection"
        }
    ]
}
```

Exportez les variables d'environnement

Exportez les variables d'environnement suivantes, de la manière appropriée pour votre système d'exploitation. Ces variables seront lues par **amqsputc** et **amqsgetc**.

Mettez à jour le chemin d'accès aux fichiers sur votre système :

```
export MQCCDTURL='<full_path_to_file>/CCDT.JSON'
export MQSSLKEYR='<full_path_to_file>/clientkey'
```

Insérez des messages dans la file d'attente

Exécutez ensuite la commande suivante :

```
amqsputc EXAMPLE.QUEUE HAEXAMPLE
```

Si la connexion au gestionnaire de files d'attente aboutit, la réponse suivante apparaît :

```
target queue is EXAMPLE.QUEUE
```

Placez plusieurs messages dans la file d'attente en entrant un texte, puis en appuyant sur **Entrée** à chaque fois.

Pour terminer, appuyez deux fois sur **Entrée**.

Extrayez les messages de la file d'attente

Exécutez ensuite la commande suivante :

```
amqsgetc EXAMPLE.QUEUE HAEXAMPLE
```

Les messages que vous avez ajoutés à l'étape précédente ont été utilisés et sont renvoyés.

Après quelques secondes, la commande prend fin.

Forcer l'échec du pod actif

Pour valider la reprise automatique du gestionnaire de files d'attente, simulez un échec du pod :

Afficher les pods actif et de secours

Exécutez ensuite la commande suivante :

```
oc get pods --selector app.kubernetes.io/instance=nativeha-example
```

Notez que dans la zone **READY** , le pod actif renvoie la valeur `1/1`, alors que les pods de réplique renvoient la valeur `0/1`.

Supprimer le pod actif

Exécutez la commande suivante en spécifiant le nom complet du pod actif :

```
oc delete pod nativeha-example-ibm-mq-<value>
```

Afficher de nouveau le statut du pod

Exécutez ensuite la commande suivante :

```
oc get pods --selector app.kubernetes.io/instance=nativeha-example
```

Afficher le statut du gestionnaire de files d'attente

Exécutez la commande suivante en spécifiant le nom complet de l'un ou l'autre des pods :

```
oc exec -t Pod -- dspmq -o nativeha -x -m HAEXAMPLE
```

Le statut doit indiquer que l'instance active a été modifiée. Par exemple :

```
QMNAME(HAEXAMPLE) ROLE(Active) INSTANCE(inst1) INSYNC(Yes) QUORUM(3/3)
INSTANCE(inst1) ROLE(Active) REPLADDR(9.20.123.45) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
INSTANCE(inst2) ROLE(Replica) REPLADDR(9.20.123.46) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
INSTANCE(inst3) ROLE(Replica) REPLADDR(9.20.123.47) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
```

Insérer et obtenir à nouveau des messages

Une fois que le pod de secours devient le pod actif (c'est-à-dire après que la valeur de la zone READY devient 1/1), utilisez à nouveau les commandes suivantes, comme décrit précédemment, pour placer des messages dans le gestionnaire de files d'attente, puis extraire des messages du gestionnaire de files d'attente :

```
amqsputc EXAMPLE.QUEUE HAEXAMPLE
```

```
amqsgetc EXAMPLE.QUEUE HAEXAMPLE
```

Félicitations, vous avez déployé un gestionnaire de files d'attente Native HA et démontré qu'il pouvait être restauré automatiquement suite à un échec de pod.

Informations supplémentaires

Remarque 1 : création d'un chemin

Si vous utilisez un IBM MQ client ou un kit d'outils antérieur à IBM MQ 9.2.1, vous devez créer un chemin.

Pour le créer, vérifiez que vous vous trouvez dans l'espace de nom que vous avez créé précédemment (voir «Avant de commencer», à la page 98), puis entrez le code YAML suivant dans la console Web Red Hat OpenShift Container Platform ou à l'aide de la ligne de commande :

```
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  name: example-mi-route
spec:
  host: hamqchl.chl.mq.ibm.com
  to:
    kind: Service
    name: nativeha-example-ibm-mq
  port:
    targetPort: 1414
  tls:
    termination: passthrough
```

Notez que le Red Hat OpenShift Container Platform Router utilise SNI pour acheminer les demandes vers le gestionnaire de files d'attente IBM MQ. Si vous modifiez le nom de canal indiqué dans la Mappe de configuration contenant des commandes MQSC, vous devez également modifier la zone hôte ici et dans le fichier CCDT.JSON qui spécifie les détails du gestionnaire de files d'attente. Pour plus d'informations, voir «Configuration d'une route pour la connexion à un gestionnaire de files d'attente depuis l'extérieur d'un cluster Red Hat OpenShift», à la page 113.

Remarque 2 : utilisation d'une classe de stockage autre que celle par défaut

Cet exemple supposant qu'une classe de stockage par défaut a été configurée dans Red Hat OpenShift Container Platform, aucune information de stockage n'est requise dans le fichier YAML des ressources personnalisées du gestionnaire de files d'attente. Si aucune classe de stockage n'est configurée par défaut ou si vous souhaitez utiliser une autre classe de stockage, ajoutez `defaultClass: <storage_class_name>` sous `spec.queueManager.storage`.

Le nom de la classe de stockage doit correspondre exactement au nom d'une classe de stockage qui existe déjà. Dans ce cas, il doit correspondre au nom renvoyé par la commande `oc get`

`storageclass`. Il doit également prendre en charge `ReadWriteMany`. Pour plus d'informations, voir «Remarques sur le stockage pour le IBM MQ Operator», à la page 11.

Tâches associées

«Affichage du statut des gestionnaires de files d'attente Native HA pour les conteneurs certifiés IBM MQ», à la page 104
Pour les conteneurs certifiés IBM MQ , vous pouvez afficher le statut des instances Native HA en exécutant la commande **dspmq** dans l'un des pods en cours d'exécution.

CP4I V 9.2.2 CD *Affichage du statut des gestionnaires de files d'attente Native HA pour les conteneurs certifiés IBM MQ*
Pour les conteneurs certifiés IBM MQ , vous pouvez afficher le statut des instances Native HA en exécutant la commande **dspmq** dans l'un des pods en cours d'exécution.

Pourquoi et quand exécuter cette tâche

Important :

Vous pouvez utiliser la commande **dspmq** dans l'un des pods en cours d'exécution pour afficher le statut opérationnel d'une instance de gestionnaire de files d'attente. Les informations renvoyées varient selon que l'instance est active ou qu'il s'agit d'une réplique. Les informations fournies par l'instance active sont définitives, tandis que celles des noeuds de réplique peuvent être obsolètes.

Vous pouvez effectuer les actions suivantes :

- Déterminer si l'instance de gestionnaire de files d'attente sur le noeud actuel est active ou s'il s'agit d'une réplique.
- Afficher le statut Native HA opérationnel de l'instance sur le noeud actuel.
- Afficher le statut opérationnel des trois instances dans une configuration Native HA.

Les zones de statut suivantes sont utilisées pour signaler le statut de la configuration Native HA :

ROLE
Indique le rôle en cours de l'instance et est l'un des rôles `Active`, `Replica` ou `Unknown`.

INSTANCE
Nom fourni pour cette instance du gestionnaire de files d'attente lorsque ce dernier a été créé à l'aide de l'option **-lr** de la commande **crtmqm**.

INSYNC
Indique si l'instance peut prendre la relève en tant qu'instance active, si nécessaire.

QUORUM
Indique le statut de quorum au format *nombre_instances_synchronisées*/*nombre_instances_configurées*.

REPLADDR
Adresse de réplication de l'instance de gestionnaire de files d'attente.

CONNACTV
Indique si le noeud est connecté à l'instance active.

BACKLOG
Indique le nombre de kilooctets de retard de l'instance.

CONNINST
Indique si l'instance désignée est connectée à cette instance.

ALTDATE
Indique la date à laquelle ces informations ont été mises à jour pour la dernière fois (vide si elles n'ont jamais été mises à jour).

ALTTIME
Indique l'heure à laquelle ces informations ont été mises à jour pour la dernière fois (vide si elles n'ont jamais été mises à jour).

Procédure

- Recherchez les pods qui font partie de votre gestionnaire de files d'attente.

```
oc get pod --selector app.kubernetes.io/instance=nativeha-qm
```

- Exécutez le dspmq dans l'un des pods

```
oc exec -t Pod dspmq
```

```
oc rsh Pod
```

pour un shell interactif, où vous pouvez exécuter dspmq directement.

- Pour déterminer si une instance de gestionnaire de files d'attente est exécutée comme instance active ou comme réplique :

```
oc exec -t Pod dspmq -o status -m QMgrName
```

Une instance active d'un gestionnaire de files d'attente nommé BOB signale le statut suivant :

```
QMNAME(BOB)                STATUS(Running)
```

Une réplique d'instance d'un gestionnaire de files d'attente nommé BOB signale le statut suivant :

```
QMNAME(BOB)                STATUS(Replica)
```

Une instance inactive signale le statut suivant :

```
QMNAME(BOB)                STATUS(Ended Immediately)
```

- Pour déterminer le statut Native HA opérationnel de l'instance dans le pod spécifié :

```
oc exec -t Pod dspmq -o nativeha -m QMgrName
```

L'instance active d'un gestionnaire de files d'attente nommé BOB peut signaler le statut suivant :

```
QMNAME(BOB)              ROLE(Active) INSTANCE(inst1) INSYNC(Yes) QUORUM(3/3)
```

Une réplique d'instance d'un gestionnaire de files d'attente nommé BOB peut signaler le statut suivant :

```
QMNAME(BOB)              ROLE(Replica) INSTANCE(inst2) INSYNC(Yes) QUORUM(2/3)
```

Une instance inactive d'un gestionnaire de files d'attente nommé BOB peut signaler le statut suivant :

```
QMNAME(BOB)              ROLE(Unknown) INSTANCE(inst3) INSYNC(no) QUORUM(0/3)
```

- Pour déterminer le statut Native HA opérationnel de toutes les instances de la configuration Native HA :

```
oc exec -t Pod dspmq -o nativeha -x -m QMgrName
```

Si vous exécutez cette commande sur le noeud qui exécute l'instance active du gestionnaire de files d'attente BOB, vous risquez de recevoir le statut suivant :

```
QMNAME(BOB)           ROLE(Active) INSTANCE(inst1) INSYNC(Yes) QUORUM(3/3)
 INSTANCE(inst1) ROLE(Active)  REPLADDR(9.20.123.45) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
 INSTANCE(inst2) ROLE(Replica) REPLADDR(9.20.123.46) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
 INSTANCE(inst3) ROLE(Replica) REPLADDR(9.20.123.47) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
```

Si vous exécutez cette commande sur un noeud qui exécute une réplique d'instance du gestionnaire de files d'attente BOB, vous risquez de recevoir le statut suivant, qui indique que l'une des répliques est en retard :

```
QMNAME(BOB)            ROLE(Replica) INSTANCE(inst2) INSYNC(Yes) QUORUM(2/3)
 INSTANCE(inst2) ROLE(Replica) REPLADDR(9.20.123.46) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
 INSTANCE(inst1) ROLE(Active)  REPLADDR(9.20.123.45) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
 INSTANCE(inst3) ROLE(Replica) REPLADDR(9.20.123.47) CONNACTV(Yes) INSYNC(No)  BACKLOG(435)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
```

Si vous exécutez cette commande sur un noeud qui exécute une instance inactive du gestionnaire de files d'attente BOB, vous risquez de recevoir le statut suivant :

```
QMNAME(BOB)            ROLE(Unknown) INSTANCE(inst3) INSYNC(no) QUORUM(0/3)
 INSTANCE(inst1) ROLE(Unknown) REPLADDR(9.20.123.45) CONNACTV(Unknown) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ALTDATE() ALTTIME()
 INSTANCE(inst2) ROLE(Unknown) REPLADDR(9.20.123.46) CONNACTV(Unknown) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ALTDATE() ALTTIME()
 INSTANCE(inst3) ROLE(Unknown) REPLADDR(9.20.123.47) CONNACTV(No) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ALTDATE() ALTTIME()
```

Si vous exécutez la commande alors que les instances sont encore en cours de négociation pour déterminer l'instance active et les répliques, vous recevez le statut suivant :

```
QMNAME(BOB)              STATUS(Negotiating)
```

Référence associée

Commande dspmq (display queue managers)

«Exemple : configuration d'un gestionnaire de files d'attente Native HA», à la page 98
Cet exemple montre comment déployer un gestionnaire de files d'attente à l'aide de la fonctionnalité de haute disponibilité native dans OCP (Red Hat OpenShift Container Platform) à l'aide d'IBM MQ Operator.

CP4I V 9.2.3 CD *Réglage avancé pour Native HA*

Paramètres avancés pour l'optimisation des délais et des intervalles. Il n'est pas nécessaire d'utiliser ces paramètres sauf si les valeurs par défaut ne respectent pas la configuration requise par votre système.

Les options de base de configuration des AP natives sont gérées à l'aide de l'API `QueueManager`, que IBM MQ Operator utilise pour configurer les fichiers INI du gestionnaire de files d'attente sous-jacent pour vous. Il existe des options plus avancées qui ne sont configurables qu'à l'aide d'un fichier INI, dans la section NativeHALocalInstance. Voir aussi «Exemple : fourniture de fichiers MQSC et INI», à la page 89 pour plus d'informations sur la configuration d'un fichier INI.

`HeartbeatInterval`
: L'intervalle des pulsations définit la fréquence en millisecondes à laquelle une instance active d'un gestionnaire de files d'attente Native HA envoie une pulsation réseau. Il est compris entre 500 (0,5 secondes) et 60000 (1 minute). Une valeur hors de cette plage empêche le démarrage du gestionnaire de files d'attente. Si cet attribut est omis, une valeur par défaut de 5000 (5 secondes) est utilisée. Chaque instance doit utiliser le même intervalle de pulsations.

`HeartbeatTimeout`
: Le dépassement du délai d'attente du signal de présence définit le temps pendant lequel une réplique d'instance d'un gestionnaire de files d'attente Native HA attend avant de considérer que l'instance active ne répondra pas. Cette valeur doit être comprise entre 500 (0,5 secondes) et 120000 (2 minutes). La valeur du dépassement du délai d'attente du signal de présence doit être supérieure ou égale à celle de l'intervalle des pulsations.

  Une valeur non valide empêche le démarrage du gestionnaire de files d'attente. Si cet attribut est omis, une réplique attend 2 x `HeartbeatInterval` avant de lancer le processus pour sélectionner une nouvelle instance active. Chaque instance doit utiliser la même valeur de dépassement du délai d'attente du signal de présence.

RetryInterval
L'intervalle entre les nouvelles tentatives définit la fréquence en millisecondes à laquelle un gestionnaire de files d'attente Native HA doit retenter un lien de réplication défectueux. Cet intervalle doit être compris entre 500 (0,5 secondes) et 120000 (2 minutes). Si cet attribut est omis, une réplique attend 2 x `HeartbeatInterval` avant de réessayer un lien de réplication ayant échoué.

CP4I *Arrêt des gestionnaires de files d'attente Native HA*

Vous pouvez utiliser la commande **endmqm** pour arrêter un gestionnaire de files d'attente actif ou de réplique faisant partie d'un groupe Native HA.

Procédure

- Pour arrêter l'instance active d'un gestionnaire de files d'attente, voir Arrêt des gestionnaires de files d'attente Native HA dans la section Configuration de cette documentation.

CP4I V 9.2.2 CD *Evaluation de la fonctionnalité Native HA dans IBM Cloud Pak for Integration 2021.1.1*

La période d'évaluation de IBM Cloud Pak for Integration 2021.1.1 Native HA est terminée. Veuillez utiliser la fonction Native HA mise à jour disponible depuis IBM Cloud Pak for Integration 2021.2.1, en utilisant IBM MQ Operator 1.6 ou version supérieure avec IBM MQ 9.2.3 ou version supérieure.

Tâches associées

«Affichage du statut des gestionnaires de files d'attente Native HA pour les conteneurs certifiés IBM MQ», à la page 104

Pour les conteneurs certifiés IBM MQ , vous pouvez afficher le statut des instances Native HA en exécutant la commande **dspmq** dans l'un des pods en cours d'exécution.

Référence associée

«Exemple : configuration d'un gestionnaire de files d'attente Native HA», à la page 98

Cet exemple montre comment déployer un gestionnaire de files d'attente à l'aide de la fonctionnalité de haute disponibilité native dans OCP (Red Hat OpenShift Container Platform) à l'aide d'IBM MQ Operator.

OpenShift CP4I ***Exemple : configuration d'un gestionnaire de files d'attente multi-instance***

Cet exemple montre comment déployer un gestionnaire de files d'attente multi-instance dans Red Hat OpenShift Container Platform (OCP) à l'aide d'IBM MQ Operator. Dans cet exemple, vous configurez également les communications TLS unidirectionnelles entre un exemple de client et le gestionnaire de files d'attente. L'exemple permet de montrer que la configuration a abouti en insérant et recevant des messages avant et après une simulation d'incident de pod.

Avant de commencer

Pour mettre en oeuvre cet exemple, vous devez d'abord avoir rempli les conditions suivantes :

- Installez IBM MQ client et ajoutez les répertoires `samp/bin` et `bin` installés à votre *CHEMIN*. Le client fournit les applications **runmqakm**, **amqsputc** et **amqsgetc** requises par cet exemple. Installez le IBM MQ client comme suit :
  - Windows Linux Pour Windows et Linux : installez le client redistribuable IBM MQ pour votre système d'exploitation à partir de https://ibm.biz/mq92redistclients
  - macOS Pour Mac : téléchargez et configurez IBM MQ MacOS Toolkit. Voir https://developer.ibm.com/tutorials/mq-macos-dev/.
- Installez l'outil OpenSSL correspondant à votre système d'exploitation. Vous en avez besoin pour générer un certificat autosigné pour le gestionnaire de files d'attente, si vous ne disposez pas déjà d'une clé privée et d'un certificat.
- Créez un projet / espace de nom Red Hat OpenShift Container Platform (OCP) pour cet exemple.
- Sur la ligne de commande, connectez-vous au cluster OCP et passez à l'espace de nom ci-dessus.

- Vérifiez que IBM MQ Operator est installé et disponible dans l'espace de nom ci-dessus.
- Configurez une classe de stockage par défaut dans OCP, à utiliser par votre gestionnaire de files d'attente. Si vous souhaitez suivre ce tutoriel sans définir de classe de stockage par défaut, reportez-vous à la rubrique Remarque 2 : utilisation d'une classe de stockage autre que celle par défaut.

A propos de cette tâche

Les gestionnaires de files d'attente multi-instance impliquent un pod actif et un pod Kubernetes de secours. Ils sont exécutés dans le cadre d'un objet StatefulSet Kubernetes avec exactement deux répliques et un ensemble de volumes persistants Kubernetes. Pour plus d'informations sur les gestionnaires de files d'attente multi-instance, reportez-vous à la rubrique «Haute disponibilité pour IBM MQ dans les conteneurs», à la page 16.

L'exemple fournit une ressource personnalisée YAML définissant un gestionnaire de files d'attente multi-instance avec stockage persistant et configuré avec TLS. Une fois que vous avez déployé le gestionnaire de files d'attente dans OCP, simulez une défaillance du pod du gestionnaire de files d'attente actif. Vous pouvez constater que la reprise automatique a été déclenchée et vérifier qu'elle a abouti en envoyant et recevant des messages après l'incident.

Exemple

Création d'une clé privée et de certificats TLS pour le serveur MQ

Cette section montre comment créer un certificat autosigné pour le gestionnaire de files d'attente et comment ajouter ce certificat à une base de données de clés pour agir en tant que fichier de clés certifiées pour le client. Si vous disposez déjà d'une clé privée et d'un certificat, vous pouvez les utiliser à la place. Notez que vous ne devez utiliser des certificats autosignés qu'à des fins de développement.

Pour créer une clé privée autosignée et un certificat public dans le répertoire de travail, exécutez la commande suivante :

```
openssl req -newkey rsa:2048 -nodes -keyout tls.key -subj "/CN=localhost" -x509 -days 3650
-out tls.crt
```

Ajout de la clé publique du gestionnaire de files d'attente à une base de données de clés client

Une base de données de clés client est utilisée comme fichier de clés certifiées pour l'application client.

Créez la base de données de clés client :

```
runmqakm -keydb -create -db clientkey.kdb -pw password -type cms -stash
```

Ajoutez la clé publique précédemment générée à la base de données de clés client :

```
runmqakm -cert -add -db clientkey.kdb -label mqservercert -file tls.crt -format ascii
-stashed
```

Création d'un secret contenant les certificats TLS pour le déploiement du gestionnaire de files d'attente

Pour que votre gestionnaire de files d'attente puisse référencer et appliquer la clé et le certificat, créez un secret TLS Kubernetes qui fait référence aux fichiers créés ci-dessus. Ce faisant, vérifiez que vous êtes dans l'espace de nom que vous avez créé avant le début de cette tâche.

```
oc create secret tls example-mi-secret --key="tls.key" --cert="tls.crt"
```

Créez une mappe de configuration contenant des commandes MQSC

Créez une mappe de configuration Kubernetes contenant les commandes MQSC pour créer une file d'attente et un canal SVRCONN, et pour ajouter un enregistrement d'authentification de canal qui autorise l'accès au canal en bloquant uniquement les utilisateurs appelés *nobody*.

Notez que cette approche ne doit être utilisée qu'à des fins de développement.

Vérifiez que vous vous trouvez dans l'espace de nom que vous avez créé précédemment (voir «Avant de commencer», à la page 107), puis entrez le code YAML suivant dans l'interface utilisateur OCP ou utilisez la ligne de commande :

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: example-mi-configmap
data:
  tls.mqsc: |
    DEFINE QLOCAL('EXAMPLE.QUEUE') DEFPSIST(YES) REPLACE
    DEFINE CHANNEL(MIQMCHL) CHLTYPE(SVRCONN) TRPTYPE(TCP) SSLCAUTH(OPTIONAL)
SSLCIPH('ANY_TLS12_OR_HIGHER')
    SET CHLAUTH(MIQMCHL) TYPE(BLOCKUSER) USERLIST('nobody') ACTION(ADD)
```

Configuration du routage

Si vous utilisez un IBM MQ client ou un kit d'outils IBM MQ 9.2.1 (ou version ultérieure), vous pouvez configurer le routage vers le gestionnaire de files d'attente à l'aide d'un fichier de configuration de gestionnaire de files d'attente (fichier INI). Dans ce fichier, vous définissez la variable *OutboundSNI* pour un routage en fonction du nom d'hôte et non du nom de canal.

Créez un fichier mqclient.ini dans le répertoire dans lequel vous exécutez les commandes. Ce fichier doit contenir le texte suivant :

```
#* Module Name: mqclient.ini                                          *#
#* Type       : IBM MQ MQI client configuration file                  *#
#  Function   : Define the configuration of a client                  *#
#*                                                                    *#
#*********************************************************************#
#* Notes      :                                                       *#
#* 1) This file defines the configuration of a client                 *#
#*                                                                    *#
#*********************************************************************#
SSL:
   OutboundSNI=HOSTNAME
```

Remarque : ne modifiez aucune valeur dans cette page. Par exemple, la chaîne HOSTNAME doit être laissée telle qu'elle est.

Pour plus d'informations, reportez-vous à la rubrique Strophe SSL du fichier de configuration client.

Si vous utilisez un IBM MQ client ou un kit d'outils antérieur à IBM MQ 9.2.1, vous devez créer un chemin OCP au lieu du fichier de configuration précédent. Suivez les étapes de la rubrique Remarque 1 : création d'un chemin.

Déployez le gestionnaire de files d'attente

Important : dans cet exemple, nous utilisons la variable *MQSNOAUT* pour désactiver l'autorisation sur le gestionnaire de files d'attente, ce qui nous permet de nous concentrer sur les étapes requises pour connecter un client à l'aide de TLS. Cette action n'est pas recommandée dans un déploiement de production d'IBM MQ, car il en résulte que toutes les applications qui se connectent disposent des pleins pouvoirs administratifs, sans mécanisme permettant de réduire les droits d'accès pour chaque application..

Copiez et mettez à jour le fichier YAML ci-après.

- Vérifiez que la licence appropriée est spécifiée. Voir Référence relative à l'octroi de licence pour mq.ibm.com/v1beta1.
- Acceptez la licence en remplaçant `false` par `true`.
- Si vous utilisez IBM Cloud File Storage, voir Remarque 3 : utilisation d'IBM Cloud File Storage

Fichier YAML des ressources personnalisées du gestionnaire de files d'attente

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: miexample
```

```
spec:
  license:
    accept: false
    license: L-RJON-C7QG3S
    use: NonProduction
  queueManager:
    name: MIEXAMPLE
    availability:
      type: MultiInstance
    mqsc:
    - configMap:
        name: example-mi-configmap
        items:
        - tls.mqsc
  template:
    pod:
      containers:
        - env:
            - name: MQSNOAUT
              value: 'yes'
          name: qmgr
  version: 9.2.5.0-r3
  web:
    enabled: true
  pki:
    keys:
      - name: example
        secret:
          secretName: example-mi-secret
          items:
          - tls.key
          - tls.crt
```

Vérifiez que vous vous trouvez dans l'espace de nom que vous avez créé précédemment, puis déployez le fichier YAML mis à jour dans l'interface utilisateur OCP, à l'aide de la ligne de commande ou à l'aide d'IBM Cloud Pak for Integration Platform Navigator.

Confirmation

Après un bref délai, le gestionnaire de files d'attente multi-instance doit être configuré et disponible. Dans cette section, nous nous assurons que le gestionnaire de files d'attente se comporte comme prévu.

Confirmez que le gestionnaire de files d'attente est en cours d'exécution

Le gestionnaire de files d'attente est en cours de déploiement. Confirmez qu'il se trouve dans l'état `Running` avant de continuer. Exemple :

```
oc get qmgr miexample
```

Testez la connexion au gestionnaire de files d'attente

Pour confirmer que le gestionnaire de files d'attente est configuré pour une communication TLS unidirectionnelle, utilisez les exemples d'applications **amqsputc** et **amqsgetc** :

Localisez le nom d'hôte du gestionnaire de files d'attente

Pour rechercher le nom d'hôte du gestionnaire de files d'attente pour la route `miexample-ibm-mq-qm`, exécutez la commande suivante. Le nom d'hôte est renvoyé dans la zone `HOST`.

```
oc get routes miexample-ibm-mq-qm
```

Spécifiez les détails du gestionnaire de files d'attente

Créez un fichier `CCDT.JSON` qui spécifie les détails du gestionnaire de files d'attente. Remplacez la valeur de l'hôte par le nom d'hôte renvoyé par l'étape précédente.

```
{
    "channel":
    [
        {
            "name": "MIQMCHL",
            "clientConnection":
            {
                "connection":
                [
                    {
```

```
                        "host": "<host from previous step>",
                        "port": 443
                    }
                ],
                "queueManager": "MIEXAMPLE"
            },
            "transmissionSecurity":
            {
              "cipherSpecification": "ECDHE_RSA_AES_128_CBC_SHA256"
            },
            "type": "clientConnection"
        }
    ]
}
```

Exportez les variables d'environnement

Exportez les variables d'environnement suivantes, de la manière appropriée pour votre système d'exploitation. Ces variables seront lues par **amqsputc** et **amqsgetc**.

Mettez à jour le chemin d'accès aux fichiers sur votre système :

```
export MQCCDTURL='<full_path_to_file>/CCDT.JSON'
export MQSSLKEYR='<full_path_to_file>/clientkey'
```

Insérez des messages dans la file d'attente

Exécutez ensuite la commande suivante :

```
amqsputc EXAMPLE.QUEUE MIEXAMPLE
```

Si la connexion au gestionnaire de files d'attente aboutit, la réponse suivante apparaît :

```
target queue is EXAMPLE.QUEUE
```

Placez plusieurs messages dans la file d'attente en entrant un texte, puis en appuyant sur **Entrée** à chaque fois.

Pour terminer, appuyez deux fois sur **Entrée**.

Extrayez les messages de la file d'attente

Exécutez ensuite la commande suivante :

```
amqsgetc EXAMPLE.QUEUE MIEXAMPLE
```

Les messages que vous avez ajoutés à l'étape précédente ont été utilisés et sont renvoyés.

Après quelques secondes, la commande prend fin.

Forcer l'échec du pod actif

Pour valider la reprise automatique du gestionnaire de files d'attente multi-instance, simulez un échec du pod :

Afficher les pods actif et de secours

Exécutez ensuite la commande suivante :

```
oc get pods
```

Notez que dans la zone **READY**, le pod actif renvoie la valeur `1/1`, alors que la pod de secours renvoie la valeur `0/1`.

Supprimer le pod actif

Exécutez la commande suivante en spécifiant le nom complet du pod actif :

```
oc delete pod miexample-ibm-mq-<value>
```

Afficher de nouveau le statut du pod

Exécutez ensuite la commande suivante :

```
oc get pods
```

Afficher le journal du pod de secours

Exécutez la commande suivante en spécifiant le nom complet du pod de secours :

```
oc logs miexample-ibm-mq-<value>
```

Vous devez voir le message suivant :

```
IBM MQ queue manager 'MIEXAMPLE' becoming the active instance.
```

Insérer et obtenir à nouveau des messages

Une fois que le pod de secours devient le pod actif (c'est-à-dire après que la valeur de la zone READY devient 1/1), utilisez à nouveau les commandes suivantes, comme décrit précédemment, pour placer des messages dans le gestionnaire de files d'attente, puis extraire des messages du gestionnaire de files d'attente :

```
amqsputc EXAMPLE.QUEUE MIEXAMPLE
```

```
amqsgetc EXAMPLE.QUEUE MIEXAMPLE
```

Félicitations, vous avez déployé un gestionnaire de files d'attente multi-instance et démontré qu'il pouvait être restauré automatiquement suite à un échec de pod.

Informations supplémentaires

Remarque 1 : création d'un chemin

Si vous utilisez un IBM MQ client ou un kit d'outils antérieur à IBM MQ 9.2.1, vous devez créer un chemin OCP.

Pour le créer, vérifiez que vous vous trouvez dans l'espace de nom que vous avez créé précédemment (voir «Avant de commencer», à la page 107), puis entrez le code YAML suivant dans l'interface utilisateur OCP ou à l'aide de la ligne de commande :

```
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  name: example-mi-route
spec:
  host: miqmchl.chl.mq.ibm.com
  to:
    kind: Service
    name: miexample-ibm-mq
  port:
    targetPort: 1414
  tls:
    termination: passthrough
```

Notez que le Red Hat OpenShift Container Platform Router utilise SNI pour acheminer les demandes vers le gestionnaire de files d'attente IBM MQ. Si vous modifiez le nom de canal indiqué dans la Mappe de configuration contenant des commandes MQSC, vous devez également modifier la zone hôte ici et dans le fichier `CCDT.JSON` qui spécifie les détails du gestionnaire de files d'attente. Pour plus d'informations, voir «Configuration d'une route pour la connexion à un gestionnaire de files d'attente depuis l'extérieur d'un cluster Red Hat OpenShift», à la page 113.

Remarque 2 : utilisation d'une classe de stockage autre que celle par défaut

Cet exemple supposant qu'une classe de stockage par défaut a été configurée dans OCP, aucune information de stockage n'est requise dans le fichier YAML des ressources personnalisées du gestionnaire de files d'attente. Si aucune classe de stockage n'est configurée par défaut ou si vous souhaitez utiliser une autre classe de stockage, ajoutez `defaultClass: <storage_class_name>` sous `spec.queueManager.storage`.

Le nom de la classe de stockage doit correspondre exactement au nom d'une classe de stockage qui existe sur votre système OCP. Dans ce cas, il doit correspondre au nom renvoyé par la commande `oc get storageclass`. Il doit également prendre en charge `ReadWriteMany`. Pour plus d'informations, voir «Remarques sur le stockage pour le IBM MQ Operator», à la page 11.

Remarque 3 : utilisation d'IBM Cloud File Storage

Dans certaines situations, par exemple, lorsque vous utilisez IBM Cloud File Storage, vous devez également spécifier la zone **securityGroups** dans Ressource personnalisée du gestionnaire de files d'attente YAML. Par exemple, en ajoutant la zone enfant suivante directement sous `spec` :

```
securityContext:
  supplementalGroups: [99]
```

Pour plus d'informations, voir «Remarques sur le stockage pour le IBM MQ Operator», à la page 11.

OpenShift CP4I CD

Configuration d'une route pour la connexion à un gestionnaire de files d'attente depuis l'extérieur d'un cluster Red Hat OpenShift

Vous avez besoin d'une route Red Hat OpenShift pour connecter une application à un gestionnaire de files d'attente IBM MQ depuis l'extérieur d'un cluster Red Hat OpenShift . Vous devez activer TLS sur votre gestionnaire de files d'attente et votre application client IBM MQ , car SNI est disponible uniquement dans le protocole TLS lorsqu'un protocole TLS 1.2 ou supérieur est utilisé. Red Hat OpenShift Container Platform Router utilise l'indication de nom de serveur pour acheminer les demandes vers le gestionnaire de files d'attente IBM MQ.

Pourquoi et quand exécuter cette tâche

Avertissement : Ce document s'applique aux versions 9.2.1 Continuous Delivery et ultérieures des clients IBM MQ. Si votre client utilise la version 9.2.0 Long Term Support ou une version antérieure, reportez-vous à la page de documentation IBM MQ 9.1 Connexion à un gestionnaire de files d'attente déployé dans un cluster Red Hat OpenShift.

V 9.2.1 La configuration requise de l' Red Hat OpenShift Route dépend du comportement de l' indication de nom de serveur (SNI) de votre application client. IBM MQ prend en charge deux paramètres d'en-tête SNI différents selon le type de configuration et de client. Un en-tête SNI est défini sur le nom d'hôte de la destination du client ou sur le nom de canal IBM MQ . Pour plus d'informations sur la façon dont IBM MQ mappe un nom de canal à un nom d'hôte, voir How IBM MQ provides multiple certificates capability.

V 9.2.1 Si un en-tête SNI est défini sur un nom de canal IBM MQ ou qu'un nom d'hôte est contrôlé à l'aide de l'attribut **OutboundSNI** . Les valeurs possibles sont `OutboundSNI=CHANNEL` (valeur par défaut) ou `OutboundSNI=HOSTNAME`. Pour plus d'informations, voir Strophe SSL du fichier de configuration du client. Notez que `CHANNEL` et `HOSTNAME` sont les valeurs exactes que vous utilisez ; il ne s'agit pas de noms de variable que vous remplacez par un nom de canal ou un nom d'hôte réel.

V 9.2.1

Comportements des clients avec différents paramètres `OutboundSNI`

Si **OutboundSNI** est défini sur `HOSTNAME`, les clients suivants définissent une SNI de nom d'hôte tant qu'un nom d'hôte est fourni dans le nom de la connexion :

- Clients C
- Clients .NET en mode non géré
- Clients Java/JMS

Si **OutboundSNI** est défini sur `HOSTNAME` et qu'une adresse IP est utilisée dans le nom de connexion, les clients suivants envoient un en-tête SNI vide :

- Clients C
- Clients .NET en mode non géré
- Clients Java/JMS (qui ne peuvent pas effectuer une recherche DNS inverse du nom d'hôte)

Si **OutboundSNI** est défini sur CHANNELou n'est pas défini, un nom de canal IBM MQ est utilisé à la place et est toujours envoyé, qu'un nom d'hôte ou un nom de connexion d'adresse IP soit utilisé ou non.

Les types de client suivants ne prennent pas en charge la définition d'un en-tête SNI dans un nom de canal IBM MQ et tentent ainsi de définir l'en-tête SNI sur un nom d'hôte quel que soit le paramètre **OutboundSNI** :

- Clients AMQP
- Clients XR
- Clients .NET en mode géré (Avant IBM MQ 9.2.0 Fix Pack 4 pour Long Term Support et avant IBM MQ 9.2.3 pour Continuous Delivery.)

V 9.2.3 V 9.2.0.4

Depuis IBM MQ 9.2.0 Fix Pack 4 for Long Term Support et IBM MQ 9.2.3 for Continuous Delivery, le client IBM MQ géré .NET a été mis à jour pour définir SERVERNAME sur le nom d'hôte respectif si la propriété **OutboundSNI** est définie sur HOSTNAME, ce qui permet à un client IBM MQ géré .NET de se connecter à un gestionnaire de files d'attente à l'aide de routes Red Hat OpenShift . Notez que, dans IBM MQ 9.2.0 Fix Pack 4, la propriété **OutboundSNI** est ajoutée et prise en charge uniquement à partir du fichier `mqclient.ini` ; vous ne pouvez pas définir la propriété à partir de l'application .NET.

V 9.2.5

Si une application client se connecte à un gestionnaire de files d'attente déployé dans un cluster Red Hat OpenShift via IBM MQ Internet Pass-Thru (MQIPT), MQIPT peut être configuré pour définir le SNI sur le nom d'hôte à l'aide de la propriété SSLClientOutboundSNI dans la définition de route.

OutboundSNI, plusieurs certificats et routes Red Hat OpenShift

IBM MQ utilise l'en-tête SNI pour fournir plusieurs fonctionnalités de certificats. Si une application se connecte à un canal IBM MQ configuré pour utiliser un certificat différent via la zone CERTLABL, elle doit se connecter avec le paramètre **OutboundSNI** de CHANNEL.

Si votre configuration de route Red Hat OpenShift requiert un HOSTNAME SNI, vous ne pouvez pas utiliser la fonctionnalité de certificats multiples de IBM MQ et vous ne pouvez pas définir de paramètre CERTLABL sur un objet canal IBM MQ .

Si une application avec un paramètre **OutboundSNI** autre que CHANNEL se connecte à un canal avec un libellé de certificat configuré, l'application est rejetée avec une erreur MQRC_SSL_INITIALIZATION_ERROR et un message AMQ9673 est imprimé dans les journaux d'erreurs du gestionnaire de files d'attente.

Pour plus d'informations sur la façon dont IBM MQ fournit plusieurs fonctionnalités de certificat, voir How IBM MQ fournit plusieurs fonctionnalités de certificat .

Exemple

Les applications client qui permettent de définir le SNI sur le canal MQ requièrent la création d'une nouvelle route Red Hat OpenShift pour chaque canal auquel vous souhaitez vous connecter. Vous devez aussi utiliser des noms de canal uniques dans votre cluster Red Hat OpenShift Container Platform pour permettre un routage vers le gestionnaire de files d'attente approprié.

Il est important que les noms de canal MQ ne se terminent pas par une lettre en minuscule, en raison de la façon dont IBM MQ associe les noms de canal aux en-têtes SNI.

Pour déterminer le nom d'hôte requis pour chacune de vos nouvelles routes Red Hat OpenShift, vous devez associer chaque nom de canal à une adresse SNI. Pour plus d'informations, voir How IBM MQ provides multiple certificates capability.

Vous devez ensuite créer une nouvelle route Red Hat OpenShift pour chaque canal, en appliquant le yaml suivant dans votre cluster :

```
  apiVersion: route.openshift.io/v1
  kind: Route
  metadata:
    name: <provide a unique name for the Route>
    namespace: <the namespace of your MQ deployment>
  spec:
    host: <SNI address mapping for the channel>
    to:
      kind: Service
      name: <the name of the Kubernetes Service for your MQ deployment (for example "<Queue Manager
Name>-ibm-mq")>
    port:
      targetPort: 1414
    tls:
      termination: passthrough
```

Configuration des détails de connexion de votre application client

Vous pouvez déterminer le nom d'hôte à utiliser pour votre connexion client en exécutant la commande suivante :

```
oc get route <Name of hostname based Route (for example "<Queue Manager Name>-ibm-mq-qm")>
-n <namespace of your MQ deployment> -o jsonpath="{.spec.host}"
```

Le port pour votre connexion client doit être le port utilisé par le routeur Red Hat OpenShift Container Platform ; il s'agit normalement du port 443.

Tâches associées

«Connexion à IBM MQ Console déployée dans un cluster Red Hat OpenShift», à la page 120
Connexion à la console IBM MQ Console d'un gestionnaire de files d'attente qui a été déployé dans un cluster Red Hat OpenShift Container Platform.

CP4I Intégration au tableau de bord des opérations d'IBM Cloud Pak for Integration

La capacité de tracer les transactions via IBM Cloud Pak for Integration est fournie par le tableau de bord des opérations.

Pourquoi et quand exécuter cette tâche

L'activation de l'intégration au tableau de bord des opérations installe un exit API MQ sur votre gestionnaire de files d'attente. L'exit API enverra au magasin de données du tableau de bord des opérations des données de trace sur les messages qui transitent par le gestionnaire de files d'attente.

Notez que seuls les messages qui sont envoyés à l'aide de liaisons client MQ sont tracés.

Notez également que pour les versions d'IBM MQ Operator antérieures à la version 1.5, si la fonction de trace est activée, les images de l'agent de trace et du collecteur déployées avec le gestionnaire de files d'attente sont toujours les dernières versions disponibles, ce qui peut introduire une incompatibilité si vous n'utilisez pas la version la plus récente d'IBM Cloud Pak for Integration.

Procédure

1. Déployez un gestionnaire de files d'attente avec la fonction de trace activée.

   Par défaut, la fonction de trace est désactivée.

   Si vous procédez au déploiement à l'aide d'IBM Cloud Pak for Integration Platform Navigator, vous pouvez activer la fonction de trace lors du déploiement en associant **Enable Tracing** à la valeur **On** et en définissant l'espace de nom dans lequel le tableau de bord des opérations est installé dans la zone **Tracing Namespace**. Pour plus d'informations sur le déploiement d'un gestionnaire de files d'attente,

voir «Déploiement d'un gestionnaire de files d'attente à l'aide d'IBM Cloud Pak for Integration Platform Navigator», à la page 85.

Si vous déployez à l'aide de Red Hat OpenShift CLI ou Console Web Red Hat OpenShift, vous pouvez activer la fonction de trace avec le fragment YAML suivant :

```
spec:
  tracing:
    enabled: true
    namespace: <Operations_Dashboard_Namespace
```

Important : le gestionnaire de files d'attente ne démarre pas tant que MQ n'est pas enregistré dans le tableau de bord des opérations (voir l'étape suivante).

Notez que lorsque la fonction est activée, elle exécute deux conteneurs sidecar ("Agent" et "Collecteur") en plus du conteneur de gestionnaire de files d'attente. Les images de ces conteneurs sidecar sont disponibles dans le même registre que l'image principale de MQ et utilisent la même stratégie d'extraction et le même secret d'extraction. Des paramètres supplémentaires sont disponibles pour configurer les limites relatives à l'unité centrale et à la mémoire.

2. S'il s'agit de la première fois qu'un gestionnaire de files d'attente d'intégration du tableau de bord des opérations a été déployé dans cet espace de nom, vous devez vous Inscrire sur le tableau de bord des opérations.

   L'enregistrement crée un objet Secret dont le pod du gestionnaire de files d'attente a besoin pour démarrer.

CP4I CD *Déploiement ou mise à niveau de IBM MQ 9.2.2 ou 9.2.3 avec l'intégration du tableau de bord des opérations dans IBM Cloud Pak for Integration 2021.4*

Chaque version de IBM MQ est associée à une version spécifique de l'agent du tableau de bord des opérations et des composants de collecteur, qui sont déployés en même temps qu'un gestionnaire de files d'attente. IBM Cloud Pak for Integration 2021.4.1 introduit une modification qui fait que les composants d'agent et de collecteur plus anciens ne fonctionnent pas avec le tableau de bord des opérations. Pour corriger cela, vous devez remplacer la version de l'agent du tableau de bord des opérations et des images de collecteur que vous utilisez lorsque vous utilisez IBM MQ 9.2.2 ou 9.2.3.

Déploiement d'un nouveau gestionnaire de files d'attente IBM MQ 9.2.2 ou 9.2.3

Lorsque vous utilisez IBM Cloud Pak for Integration 2021.4.1 avec IBM MQ 9.2.2 ou 9.2.3, vous devez remplacer l'agent tableau de bord des opérations et les images de collecteur dans les versions 2.4 de votre `QueueManager` YAML. Exemple :

```
spec:
  tracing:
    agent:
      image: cp.icr.io/cp/icp4i/od/icp4i-od-
agent@sha256:27a211f0f78eff765d1f9520e0f9841f902600bb556827477b206e209cb44d20
    collector:
      image: cp.icr.io/cp/icp4i/od/icp4i-od-
collector@sha256:dc70b1341b23dc72642ce68809811f9db0e8a0c46bda2508e8eb3d4035e04f4b
```

Si vous ne le faites pas, votre Pod `QueueManager` sera bloqué dans l'état `Pending`. Lorsque vous effectuez une mise à niveau vers IBM MQ 9.2.4, vous pouvez supprimer ces substitutions.

Mise à niveau vers IBM Cloud Pak for Integration 2021.4.1

Remarque : Si vous conservez votre gestionnaire de files d'attente IBM MQ 9.2.2 ou 9.2.3, n'effectuez pas l'étape 3.

1. Mettez à jour `QueueManager` pour remplacer les images de l'agent et du collecteur, comme indiqué précédemment.

2. Mettez à niveau vos opérateurs IBM Cloud Pak for Integration, y compris le tableau de bord des opérations et l'opérateur IBM MQ, comme décrit dans «Mise à niveau de IBM MQ Operator et des gestionnaires de files d'attente», à la page 74.
3. (facultatif) Pour effectuer une mise à niveau vers IBM MQ 9.2.4 ou une version ultérieure, mettez à jour `QueueManager` pour utiliser `.spec.version` pour votre version de IBM MQ, puis supprimez la substitution des images de l'agent et du collecteur.

OpenShift CP4I Génération d'une image avec des fichiers MQSC et INI personnalisés, à l'aide de l'interface de ligne de commande Red Hat OpenShift

Utilisez un pipeline Red Hat OpenShift Container Platform pour créer une image de conteneur IBM MQ, avec les fichiers MQSC et INI que vous souhaitez appliquer aux gestionnaires de files d'attente à l'aide de cette image. Cette tâche doit être effectuée par un administrateur de projet.

Avant de commencer

Vous devez installer l'interface de ligne de commande Red Hat OpenShift Container Platform.

Connectez-vous à votre cluster avec **`cloudctl login`** (pour IBM Cloud Pak for Integration) ou **`oc login`**.

Si vous ne disposez pas de Red Hat OpenShift secret pour le registre autorisé IBM dans votre projet Red Hat OpenShift, suivez la procédure de «Préparation de votre projet Red Hat OpenShift pour IBM MQ», à la page 83.

Procédure

1. Créez un`ImageStream`

   Un flux d'image et les étiquettes qui lui sont associées fournissent une abstraction pour les images de conteneur de référence depuis Red Hat OpenShift Container Platform. Ils vous permettent de savoir quelles images sont disponibles et de vous assurer que vous utilisez l'image spécifique dont vous avez besoin, même si l'image dans le référentiel change.

   ```
   oc create imagestream mymq
   ```

2. Créer un `BuildConfig` pour votre nouvelle image

   Un `BuildConfig` permet de créer pour votre nouvelle image, qui sera basée sur les images officielles IBM, mais qui ajoutera tous les fichiers MQSC ou INI que vous souhaitez exécuter sur le démarrage du conteneur.

   a) Créez un fichier YAML définissant la ressource `BuildConfig`

   Par exemple, créez un fichier nommé "mq-build-config.yaml" dont le contenu est le suivant :

   ```
   apiVersion: build.openshift.io/v1
   kind: BuildConfig
   metadata:
     name: mymq
   spec:
     source:
       dockerfile: |-
         FROM cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.5.0-r3
         RUN printf "DEFINE QLOCAL(foo) REPLACE\n" > /etc/mqm/my.mqsc \
           && printf "Channels:\n\tMQIBindType=FASTPATH\n" > /etc/mqm/my.ini
         LABEL summary "My custom MQ image"
     strategy:
       type: Docker
       dockerStrategy:
         from:
           kind: "DockerImage"
           name: "cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.5.0-r3"
         pullSecret:
           name: ibm-entitlement-key
     output:
   ```

```
    to:
      kind: ImageStreamTag
      name: 'mymq:latest-amd64'
```

Vous devrez remplacer les deux emplacements du produit IBM MQ de base afin de désigner l'image de base appropriée pour la version et le correctif que vous voulez utiliser (voir «Historique des éditions de IBM MQ Operator», à la page 21 pour plus de détails). Au fur et à mesure que les correctifs sont appliqués, vous devez répéter ces étapes afin de régénérer votre image.

Cet exemple crée une nouvelle image basée sur l'image officielle IBM et ajoute les fichiers "my.mqsc" et "my.ini" dans le répertoire `/etc/mqm`. Tout fichier MQSC ou INI trouvé dans ce répertoire sera appliqué par le conteneur au démarrage. Les fichiers INI sont appliqués avec l'option **`crtmqm -ii`** et fusionnés avec les fichiers INI existants. Les fichiers MQSC sont appliqués par ordre alphabétique.

Il est important que vos commandes MQSC puissent être réexécutées, car elles seront exécutées *à chaque fois* que le gestionnaire de files d'attente démarre. Cela implique généralement d'ajouter le paramètre REPLACE à toutes les commandes DEFINE et d'ajouter le paramètre `IGNSTATE(YES)` à toutes les commandes START ou STOP.

b) Appliquez le `BuildConfig` au serveur.

```
oc apply -f mq-build-config.yaml
```

3. Exécutez une génération pour créer votre image.

a) Démarrez la génération.

```
oc start-build mymq
```

Une sortie similaire à la suivante apparaît :

```
build.build.openshift.io/mymq-1 started
```

b) Vérifiez le statut de la génération.
Par exemple, vous pouvez exécuter la commande suivante en utilisant l'identificateur de génération renvoyé à l'étape précédente :

```
oc describe build mymq-1
```

4. Déployez un gestionnaire de files d'attente en utilisant votre nouvelle image.

Suivez les étapes décrites dans la rubrique «Déploiement d'un gestionnaire de files d'attente sur un cluster Red Hat OpenShift Container Platform», à la page 85 pour ajouter votre nouvelle image personnalisée dans le fichier YAML.

Vous pouvez ajouter le fragment suivant de YAML dans votre YAML `QueueManager` normal, où *SingleNamespace* correspond au projet / espace de nom Red Hat OpenShift que vous utilisez, et *Image* est le nom de l'image que vous avez créée précédemment (par exemple, "mymq:latest-amd64") :

```
spec:
  queueManager:
    image: image-registry.openshift-image-registry.svc:5000/my-namespace/my-image
```

Tâches associées

«Déploiement d'un gestionnaire de files d'attente sur un cluster Red Hat OpenShift Container Platform», à la page 85
Utilisez la ressource personnalisée QueueManager pour déployer un gestionnaire de files d'attente sur un cluster Red Hat OpenShift Container Platform.

OpenShift CP4I Ajout d'annotations et d'étiquettes personnalisées aux ressources du gestionnaire de files d'attente

Vous ajoutez des annotations et des étiquettes personnalisées aux métadonnées QueueManager.

Pourquoi et quand exécuter cette tâche

Les annotations et les étiquettes personnalisées sont ajoutées à toutes les ressources, à l'exception des PVC. Si une annotation ou une étiquette personnalisée correspond à une clé existante, la valeur définie par IBM MQ Operator est utilisée.

Procédure

- Ajoutez des annotations personnalisées.

  Pour ajouter des annotations personnalisées aux ressources du gestionnaire de files d'attente, y compris le pod, ajoutez les annotations sous `metadata`. Exemple :

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: quickstart-cp4i
  annotations:
    annotationKey: "value"
```

- Ajoutez des étiquettes personnalisées.

  Pour ajouter des étiquettes personnalisées aux ressources du gestionnaire de files d'attente, y compris le pod, ajoutez les étiquettes sous `metadata`. Exemple :

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: quickstart-cp4i
  labels:
    labelKey: "value"
```

OpenShift CP4I Désactivation des vérifications des webhooks d'exécution

Les vérifications des webhooks d'exécution garantissent que les classes de stockage sont viables pour votre gestionnaire de files d'attente. Vous les désactivez pour améliorer les performances, ou parce qu'elles ne sont pas valides pour votre environnement.

Pourquoi et quand exécuter cette tâche

Les vérifications des webhooks d'exécution sont effectuées sur la configuration du gestionnaire de files d'attente. Elles garantissent que les classes de stockage conviennent au type du gestionnaire de files d'attente sélectionné.

Vous pouvez choisir de désactiver ces vérifications pour réduire le temps de création du gestionnaire de files d'attente ou parce que les vérifications ne sont pas valides pour votre environnement spécifique.

Remarque : Lorsque vous désactivez les vérifications des webhooks d'exécution, toutes les valeurs de classe de stockage sont autorisées. Cela peut entraîner une rupture de gestionnaire de files d'attente.

La prise en charge des vérifications d'exécution a été introduite dans IBM MQ Operator 1.2.

Procédure

- Désactivez les vérifications des webhooks d'exécution.

  Ajoutez l'annotation suivante sous `metadata`. Exemple :

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: quickstart-cp4i
  annotations:
    "com.ibm.cp4i/disable-webhook-runtime-checks" : "true"
```

OpenShift CP4I Utilisation de IBM MQ à l'aide de IBM MQ Operator

Pourquoi et quand exécuter cette tâche

Procédure

- «Préparation de votre projet Red Hat OpenShift pour IBM MQ», à la page 83.
- «Déploiement d'un gestionnaire de files d'attente sur un cluster Red Hat OpenShift Container Platform», à la page 85.

OpenShift CP4I Connexion à IBM MQ Console déployée dans un cluster Red Hat OpenShift

Connexion à la console IBM MQ Console d'un gestionnaire de files d'attente qui a été déployé dans un cluster Red Hat OpenShift Container Platform.

Pourquoi et quand exécuter cette tâche

L'URL IBM MQ Console est disponible dans la page des détails `QueueManager` de la console Web Red Hat OpenShift ou dans le IBM Cloud Pak for Integration Platform Navigator. Vous pouvez également la trouver à partir de l'interface de ligne de commande Red Hat OpenShift en exécutant la commande suivante :

```
oc get queuemanager <QueueManager Name> -n <namespace of your MQ deployment> --output
jsonpath='{.status.adminUiUrl}'
```

Si vous utilisez une licence IBM Cloud Pak for Integration, la console Web IBM MQ est configurée pour utiliser IBM Cloud Pak Identity and Access Manager (IAM). Le composant IAM a peut-être été déjà configuré par votre administrateur de cluster. Toutefois, s'il s'agit de la première utilisation d'IAM sur votre cluster Red Hat OpenShift, vous devez récupérer le mot de passe administrateur initial. Pour plus d'informations, voir Getting the initial admin password.

Si vous utilisez une licence IBM MQ, la console Web MQ n'est pas préconfigurée et vous devez la configurer vous-même. Pour plus d'informations, voir Configuration des utilisateurs et des rôles.

Tâches associées

«Configuration d'une route pour la connexion à un gestionnaire de files d'attente depuis l'extérieur d'un cluster Red Hat OpenShift», à la page 113
Vous avez besoin d'une route Red Hat OpenShift pour connecter une application à un gestionnaire de files d'attente IBM MQ depuis l'extérieur d'un cluster Red Hat OpenShift . Vous devez activer TLS sur votre gestionnaire de files d'attente et votre application client IBM MQ , car SNI est disponible uniquement dans le protocole TLS lorsqu'un protocole TLS 1.2 ou supérieur est utilisé. Red Hat OpenShift Container Platform Router utilise l'indication de nom de serveur pour acheminer les demandes vers le gestionnaire de files d'attente IBM MQ.

OpenShift CP4I *Octroi de droits pour le IBM MQ Console à l'aide d' IBM Cloud Pak IAM*

Les droits de l' IBM MQ Console sont gérés via le concentrateur d'administration IBM Cloud Pak , et non via IBM Cloud Pak for Integration Platform Navigator. IBM MQ n'utilise pas les droits d'automatisation fournis par IBM Cloud Pak for Integration, mais utilise les droits de base activés par IBM Cloud Pak Identity and Access Manager (IAM).

Procédure

1. Ouvrez la console d'administration IBM Cloud Pak.

   Dans le IBM Cloud Pak for Integration Platform UI, cliquez sur le commutateur Cloud Pak (icône à 9 points) dans l'angle supérieur droit de la barre d'outils, puis cliquez sur le panneau **IBM Cloud Pak Administration** .

2. Dans le menu de navigation dans l'angle supérieur gauche, sélectionnez **Identité et accès**, puis sélectionnez **Equipes et ID de services**.
3. Créez une équipe, puis ajoutez-y des utilisateurs.
   a) Sélectionnez **Créer une équipe**.
   b) Entrez un nom d'équipe, puis sélectionnez le domaine de sécurité pour les utilisateurs que vous souhaitez gérer.
   c) Recherchez des utilisateurs.
      Ces utilisateurs doivent déjà exister dans votre fournisseur d'identité.
   d) Lorsque vous trouvez chaque utilisateur, attribuez-lui un rôle. Il doit s'agir de "Administrator" ou de "Cluster Administrator", pour administrer IBM MQ à l'aide de IBM MQ Console.
4. Ajoutez chaque utilisateur à un espace de nom.
   a) Sélectionnez l'équipe pour l'éditer.
   b) Sélectionnez **Ressources** > **Gérer les ressources**.
   c) Sélectionnez les espaces de nom que cette équipe doit administrer. Il peut s'agir de n'importe quel espace de nom avec un gestionnaire de files d'attente.

OpenShift CP4I Surveillance lors de l'utilisation de IBM MQ Operator

Les gestionnaires de files d'attente gérés par le IBM MQ Operator peuvent produire des métriques compatibles avec Prometheus.

Vous pouvez afficher ces métriques à l'aide de la pile de surveillanceRed Hat OpenShift Container Platform (OCP). Ouvrez l'onglet **Métriques** dans OCP, puis cliquez sur **Observe** > **Métriques**. Les métriques de gestionnaire de files d'attente sont activées par défaut, mais peuvent être désactivées en définissant **`.spec.metrics.enabled`** sur `false`.

Prometheus est une base de données de série temporelle et un moteur d'évaluation de règle pour les métriques. Les conteneurs IBM MQ exposent un nœud final de métriques qui peut être interrogé par Prometheus. Les métriques sont générées à partir des rubriques du système MQ pour la surveillance et la trace d'activité.

Red Hat OpenShift Container Platform inclut une pile de surveillance préconfigurée, préinstallée et mise à jour automatiquement qui utilise un serveur Prometheus. La pile de surveillance Red Hat OpenShift Container Platform doit être configurée pour surveiller les projets définis par l'utilisateur. Pour plus d'informations, voir Enabling monitoring for user-defined projects. IBM MQ Operator crée un `ServiceMonitor` lorsque vous créez un `QueueManager` avec les mesures activées, que l'opérateur Prometheus peut ensuite reconnaître.

Dans les versions plus anciennes d'IBM Cloud Pak for Integration, vous pouvez également utiliser le service IBM Cloud Platform Monitoring pour fournir un serveur Prometheus à la place.

OpenShift CP4I *Métriques publiées lors de l'utilisation de IBM MQ Operator*

Les conteneurs du gestionnaire de files d'attente peuvent publier des mesures compatibles avec le monitoring Red Hat OpenShift.

Métrique	Tapez	Description
`ibmmq_qmgr_commit_total`	`counter`	Nombre de validations
`ibmmq_qmgr_cpu_load_fifteen_minute_average_percentage`	`gauge`	Charge UC - moyenne sur quinze minutes
`ibmmq_qmgr_cpu_load_five_minute_average_percentage`	`gauge`	Charge UC - moyenne sur cinq minutes

Métrique	Tapez	Description
ibmmq_qmgr_cpu_load_one_minute_average_percentage	gauge	Charge UC - moyenne sur une minute
ibmmq_qmgr_destructive_get_bytes_total	counter	Commande get destructive d'intervalle total - nombre d'octets
ibmmq_qmgr_destructive_get_total	counter	Commande get destructive d'intervalle total - nombre
ibmmq_qmgr_durable_subscription_alter_total	counter	Modification du nombre d'abonnements durables
ibmmq_qmgr_durable_subscription_create_total	counter	Création du nombre d'abonnements durables
ibmmq_qmgr_durable_subscription_delete_total	counter	Suppression du nombre d'abonnements durables
ibmmq_qmgr_durable_subscription_resume_total	counter	Reprise du nombre d'abonnements durables
ibmmq_qmgr_errors_file_system_free_space_percentage	gauge	Système de fichiers d'erreurs MQ - espace disponible
ibmmq_qmgr_errors_file_system_in_use_bytes	gauge	Système de fichiers d'erreurs MQ - octets utilisés
ibmmq_qmgr_expired_message_total	counter	Nombre de messages arrivés à expiration
ibmmq_qmgr_failed_browse_total	counter	Echec du calcul du nombre de visualisation
ibmmq_qmgr_failed_mqcb_total	counter	Echec du calcul du nombre de MQCB
ibmmq_qmgr_failed_mqclose_total	counter	Echec du calcul du nombre de MQCLOSE
ibmmq_qmgr_failed_mqconn_mqconnx_total	counter	Echec du calcul du nombre de MQCONN/ MQCONNX
ibmmq_qmgr_failed_mqget_total	counter	Echec de MQGET - nombre
ibmmq_qmgr_failed_mqinq_total	counter	Echec du calcul du nombre de MQINQ
ibmmq_qmgr_failed_mqopen_total	counter	Echec du calcul du nombre de MQOPEN
ibmmq_qmgr_failed_mqput1_total	counter	Echec du calcul du nombre de MQPUT1

Métrique	Tapez	Description
ibmmq_qmgr_failed_mqput_total	counter	Echec du calcul du nombre de MQPUT
ibmmq_qmgr_failed_mqset_total	counter	Echec du calcul du nombre de MQSET
ibmmq_qmgr_failed_mqsubrq_total	counter	Echec du calcul du nombre de MQSUBRQ
ibmmq_qmgr_failed_subscription_create_alter_resume_total	counter	Echec de la création/modification/reprise du calcul du nombre d'abonnements
ibmmq_qmgr_failed_subscription_delete_total	counter	Nombre d'échec de suppression d'abonnement
ibmmq_qmgr_failed_topic_mqput_mqput1_total	counter	Echec du calcul du nombre de MQPUT/MQPUT1 de rubrique
ibmmq_qmgr_fdc_files	gauge	Nombre de fichiers FDC MQ
ibmmq_qmgr_log_file_system_in_use_bytes	gauge	Système de fichier journal - octets utilisés
ibmmq_qmgr_log_file_system_max_bytes	gauge	Système de fichier journal - nombre d'octets maximal
ibmmq_qmgr_log_in_use_bytes	gauge	Journal - octets utilisés
ibmmq_qmgr_log_logical_written_bytes_total	counter	Journal - octets logiques écrits
ibmmq_qmgr_log_max_bytes	gauge	Journal - nombre d'octets maximal
ibmmq_qmgr_log_physical_written_bytes_total	counter	Journal - octets physiques écrits
ibmmq_qmgr_log_primary_space_in_use_percentage	gauge	Journal - espace principal en cours utilisé
ibmmq_qmgr_log_workload_primary_space_utilization_percentage	gauge	Journal - utilisation de l'espace principal de charge de travail
ibmmq_qmgr_log_write_latency_seconds	gauge	Journal - temps d'attente d'écriture
ibmmq_qmgr_log_write_size_bytes	gauge	Journal - taille d'écriture

Métrique	Tapez	Description
ibmmq_qmgr_mqcb_total	counter	Nombre de MQCB
ibmmq_qmgr_mqclose_total	counter	Nombre de MQCLOSE
ibmmq_qmgr_mqconn_mqconnx_total	counter	Nombre de MQCONN/MQCONNX
ibmmq_qmgr_mqctl_total	counter	Nombre de MQCTL
ibmmq_qmgr_mqdisc_total	counter	Nombre de MQDISC
ibmmq_qmgr_mqinq_total	counter	Nombre de MQINQ
ibmmq_qmgr_mqopen_total	counter	Nombre de MQOPEN
ibmmq_qmgr_mqput_mqput1_bytes_total	counter	Nombre d'octets MQPUT/MQPUT1 d'intervalle total
ibmmq_qmgr_mqput_mqput1_total	counter	Nombre de MQPUT/MQPUT1 d'intervalle total
ibmmq_qmgr_mqset_total	counter	Nombre de MQSET
ibmmq_qmgr_mqstat_total	counter	Nombre de MQSTAT
ibmmq_qmgr_mqsubrq_total	counter	Nombre de MQSUBRQ
ibmmq_qmgr_non_durable_subscription_create_total	counter	Création du nombre d'abonnements non durables
ibmmq_qmgr_non_durable_subscription_delete_total	counter	Suppression du nombre d'abonnements non durables
ibmmq_qmgr_non_persistent_message_browse_bytes_total	counter	Visualisation de messages non persistants - nombre d'octets
ibmmq_qmgr_non_persistent_message_browse_total	counter	Visualisation de messages non persistants - nombre
ibmmq_qmgr_non_persistent_message_destructive_get_total	counter	Commande get destructive de message non persistant - nombre
ibmmq_qmgr_non_persistent_message_get_bytes_total	counter	Messages non persistants obtenus - nombre d'octets

Métrique	Tapez	Description
ibmmq_qmgr_non_persistent_message_mqput1_total	counter	Nombre de MQPUT1 de message non persistant
ibmmq_qmgr_non_persistent_message_mqput_total	counter	Nombre de MQPUT de message non persistant
ibmmq_qmgr_non_persistent_message_put_bytes_total	counter	Messages non persistants insérés - nombre d'octets
ibmmq_qmgr_non_persistent_topic_mqput_mqput1_total	counter	Non persistant - nombre de MQPUT/MQPUT1 de rubrique
ibmmq_qmgr_persistent_message_browse_bytes_total	counter	Visualisation de messages persistants - nombre d'octets
ibmmq_qmgr_persistent_message_browse_total	counter	Visualisation de messages persistants - nombre
ibmmq_qmgr_persistent_message_destructive_get_total	counter	Commande get destructive de message persistant - nombre
ibmmq_qmgr_persistent_message_get_bytes_total	counter	Messages persistants obtenus - nombre d'octets
ibmmq_qmgr_persistent_message_mqput1_total	counter	Nombre de MQPUT1 de message persistant
ibmmq_qmgr_persistent_message_mqput_total	counter	Nombre de MQPUT de message persistant
ibmmq_qmgr_persistent_message_put_bytes_total	counter	Messages persistants insérés - nombre d'octets
ibmmq_qmgr_persistent_topic_mqput_mqput1_total	counter	Persistant - nombre de MQPUT/MQPUT1 de rubrique
ibmmq_qmgr_published_to_subscribers_bytes_total	counter	Publication aux abonnés - nombre d'octets
ibmmq_qmgr_published_to_subscribers_message_total	counter	Publication aux abonnés - nombre de messages
ibmmq_qmgr_purged_queue_total	counter	Nombre de files d'attente purgées

Métrique	Tapez	Description
ibmmq_qmgr_queue_manager_file_system_free_space_percentage	gauge	Système de fichiers du gestionnaire de files d'attente - espace disponible
ibmmq_qmgr_queue_manager_file_system_in_use_bytes	gauge	Système de fichiers du gestionnaire de files d'attente - octets utilisés
ibmmq_qmgr_ram_free_percentage	gauge	Pourcentage de mémoire vive disponible
ibmmq_qmgr_ram_usage_estimate_for_queue_manager_bytes	gauge	Nombre total d'octets de mémoire vive - estimation pour le gestionnaire de files d'attente
ibmmq_qmgr_rollback_total	counter	Nombre d'annulations
ibmmq_qmgr_system_cpu_time_estimate_for_queue_manager_percentage	gauge	Temps UC système - estimation du pourcentage pour le gestionnaire de files d'attente
ibmmq_qmgr_system_cpu_time_percentage	gauge	Pourcentage de temps UC système
ibmmq_qmgr_topic_mqput_mqput1_total	counter	Intervalle total MQPUT/MQPUT1 de rubrique
ibmmq_qmgr_topic_put_bytes_total	counter	Octets de rubrique d'intervalle total insérés
ibmmq_qmgr_trace_file_system_free_space_percentage	gauge	Système de fichiers de trace MQ - espace disponible
ibmmq_qmgr_trace_file_system_in_use_bytes	gauge	Système de fichiers de trace MQ - octets utilisés
ibmmq_qmgr_user_cpu_time_estimate_for_queue_manager_percentage	gauge	Temps UC utilisateur - estimation du pourcentage pour le gestionnaire de files d'attente
ibmmq_qmgr_user_cpu_time_percentage	gauge	Pourcentage de temps UC utilisateur

CP4I V 9.2.2 CD

Affichage du statut des gestionnaires de files d'attente Native HA pour les conteneurs certifiés IBM MQ

Pour les conteneurs certifiés IBM MQ , vous pouvez afficher le statut des instances Native HA en exécutant la commande **dspmq** dans l'un des pods en cours d'exécution.

Pourquoi et quand exécuter cette tâche

Important :

Vous pouvez utiliser la commande **dspmq** dans l'un des pods en cours d'exécution pour afficher le statut opérationnel d'une instance de gestionnaire de files d'attente. Les informations renvoyées varient selon que l'instance est active ou qu'il s'agit d'une réplique. Les informations fournies par l'instance active sont définitives, tandis que celles des noeuds de réplique peuvent être obsolètes.

Vous pouvez effectuer les actions suivantes :

- Déterminer si l'instance de gestionnaire de files d'attente sur le noeud actuel est active ou s'il s'agit d'une réplique.
- Afficher le statut Native HA opérationnel de l'instance sur le noeud actuel.
- Afficher le statut opérationnel des trois instances dans une configuration Native HA.

Les zones de statut suivantes sont utilisées pour signaler le statut de la configuration Native HA :

ROLE
Indique le rôle en cours de l'instance et est l'un des rôles `Active`, `Replica` ou `Unknown`.

INSTANCE
Nom fourni pour cette instance du gestionnaire de files d'attente lorsque ce dernier a été créé à l'aide de l'option **-lr** de la commande **crtmqm**.

INSYNC
Indique si l'instance peut prendre la relève en tant qu'instance active, si nécessaire.

QUORUM
Indique le statut de quorum au format *nombre_instances_synchronisées*/*nombre_instances_configurées*.

REPLADDR
Adresse de réplication de l'instance de gestionnaire de files d'attente.

CONNACTV
Indique si le noeud est connecté à l'instance active.

BACKLOG
Indique le nombre de kilooctets de retard de l'instance.

CONNINST
Indique si l'instance désignée est connectée à cette instance.

ALTDATE
Indique la date à laquelle ces informations ont été mises à jour pour la dernière fois (vide si elles n'ont jamais été mises à jour).

ALTTIME
Indique l'heure à laquelle ces informations ont été mises à jour pour la dernière fois (vide si elles n'ont jamais été mises à jour).

Procédure

- Recherchez les pods qui font partie de votre gestionnaire de files d'attente.

```
oc get pod --selector app.kubernetes.io/instance=nativeha-qm
```

- Exécutez le `dspmq` dans l'un des pods

```
oc exec -t Pod dspmq
```

```
oc rsh Pod
```

pour un shell interactif, où vous pouvez exécuter `dspmq` directement.

- Pour déterminer si une instance de gestionnaire de files d'attente est exécutée comme instance active ou comme réplique :

```
oc exec -t Pod dspmq -o status -m QMgrName
```

Une instance active d'un gestionnaire de files d'attente nommé BOB signale le statut suivant :

```
QMNAME(BOB)                STATUS(Running)
```

Une réplique d'instance d'un gestionnaire de files d'attente nommé BOB signale le statut suivant :

```
QMNAME(BOB)                STATUS(Replica)
```

Une instance inactive signale le statut suivant :

```
QMNAME(BOB)                STATUS(Ended Immediately)
```

- Pour déterminer le statut Native HA opérationnel de l'instance dans le pod spécifié :

```
oc exec -t Pod dspmq -o nativeha -m QMgrName
```

L'instance active d'un gestionnaire de files d'attente nommé BOB peut signaler le statut suivant :

```
QMNAME(BOB)              ROLE(Active) INSTANCE(inst1) INSYNC(Yes) QUORUM(3/3)
```

Une réplique d'instance d'un gestionnaire de files d'attente nommé BOB peut signaler le statut suivant :

```
QMNAME(BOB)              ROLE(Replica) INSTANCE(inst2) INSYNC(Yes) QUORUM(2/3)
```

Une instance inactive d'un gestionnaire de files d'attente nommé BOB peut signaler le statut suivant :

```
QMNAME(BOB)              ROLE(Unknown) INSTANCE(inst3) INSYNC(no) QUORUM(0/3)
```

- Pour déterminer le statut Native HA opérationnel de toutes les instances de la configuration Native HA :

```
oc exec -t Pod dspmq -o nativeha -x -m QMgrName
```

Si vous exécutez cette commande sur le noeud qui exécute l'instance active du gestionnaire de files d'attente BOB, vous risquez de recevoir le statut suivant :

```
QMNAME(BOB)            ROLE(Active) INSTANCE(inst1) INSYNC(Yes) QUORUM(3/3)
 INSTANCE(inst1) ROLE(Active)  REPLADDR(9.20.123.45) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
 INSTANCE(inst2) ROLE(Replica) REPLADDR(9.20.123.46) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
 INSTANCE(inst3) ROLE(Replica) REPLADDR(9.20.123.47) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
```

Si vous exécutez cette commande sur un noeud qui exécute une réplique d'instance du gestionnaire de files d'attente BOB, vous risquez de recevoir le statut suivant, qui indique que l'une des répliques est en retard :

```
QMNAME(BOB)            ROLE(Replica) INSTANCE(inst2) INSYNC(Yes) QUORUM(2/3)
 INSTANCE(inst2) ROLE(Replica) REPLADDR(9.20.123.46) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
 INSTANCE(inst1) ROLE(Active)  REPLADDR(9.20.123.45) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
 INSTANCE(inst3) ROLE(Replica) REPLADDR(9.20.123.47) CONNACTV(Yes) INSYNC(No)  BACKLOG(435)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
```

Si vous exécutez cette commande sur un noeud qui exécute une instance inactive du gestionnaire de files d'attente BOB, vous risquez de recevoir le statut suivant :

```
QMNAME(BOB)            ROLE(Unknown) INSTANCE(inst3) INSYNC(no) QUORUM(0/3)
 INSTANCE(inst1) ROLE(Unknown) REPLADDR(9.20.123.45) CONNACTV(Unknown) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ALTDATE() ALTTIME()
 INSTANCE(inst2) ROLE(Unknown) REPLADDR(9.20.123.46) CONNACTV(Unknown) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ALTDATE() ALTTIME()
 INSTANCE(inst3) ROLE(Unknown) REPLADDR(9.20.123.47) CONNACTV(No) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ALTDATE() ALTTIME()
```

Si vous exécutez la commande alors que les instances sont encore en cours de négociation pour déterminer l'instance active et les répliques, vous recevez le statut suivant :

```
QMNAME(BOB)             STATUS(Negotiating)
```

Référence associée

Commande dspmq (display queue managers)

«Exemple : configuration d'un gestionnaire de files d'attente Native HA», à la page 98
Cet exemple montre comment déployer un gestionnaire de files d'attente à l'aide de la fonctionnalité de haute disponibilité native dans OCP (Red Hat OpenShift Container Platform) à l'aide d'IBM MQ Operator.

OpenShift CP4I Sauvegarde et restauration de la configuration du gestionnaire de files d'attente à l'aide de l'interface de ligne de commande Red Hat OpenShift

Effectuez une sauvegarde de la configuration de gestionnaire de files d'attente pour pouvoir régénérer un gestionnaire de files d'attente depuis ses définitions en cas de perte de la configuration de gestionnaire de files d'attente. Cette procédure n'effectue pas de sauvegarde des données de journal du gestionnaire de files d'attente. En raison de la nature transitoire des messages, les données de journal historiques ne sont généralement pas pertinentes au moment de la restauration.

Avant de commencer

Connectez-vous à votre cluster avec **cloudctl login** (pour IBM Cloud Pak for Integration) ou **oc login**.

Procédure

- Effectuez une sauvegarde de la configuration de gestionnaire de files d'attente.

  Vous pouvez utiliser la commande **dmpmqcfg** pour vider la configuration d'un gestionnaire de files d'attente IBM MQ.

  a) Obtenez le nom du pod pour votre gestionnaire de files d'attente.
  Par exemple, vous pouvez exécuter la commande suivante, où *nom_gestionnaire_files_attente* est le nom de votre ressource `QueueManager` :

  ```
  oc get pods --selector app.kubernetes.io/name=ibm-mq,app.kubernetes.io/
  instance=queue_manager_name
  ```

  b) Exécutez la commande **dmpmqcfg** sur le pod, en dirigeant la sortie dans un fichier sur votre machine locale.

  dmpmqcfg génère la configuration MQSC du gestionnaire de files d'attente.

  ```
  oc exec -it pod_name -- dmpmqcfg > backup.mqsc
  ```

- Restaurez la configuration de gestionnaire de files d'attente.

  Après avoir suivi la procédure de sauvegarde décrite à l'étape précédente, vous devez disposer d'un fichier `backup.mqsc` contenant la configuration du gestionnaire de files d'attente. Vous pouvez restaurer la configuration en appliquant ce fichier à un nouveau gestionnaire de files d'attente.

  a) Obtenez le nom du pod pour votre gestionnaire de files d'attente.
  Par exemple, vous pouvez exécuter la commande suivante, où *nom_gestionnaire_files_attente* est le nom de votre ressource `QueueManager` :

  ```
  oc get pods --selector app.kubernetes.io/name=ibm-mq,app.kubernetes.io/
  instance=queue_manager_name
  ```

  b) Exécutez la commande **runmqsc** sur le pod, en dirigeant la sortie dans le fichier `backup.mqsc`.

```
oc exec -i pod_name -- runmqsc < backup.mqsc
```

OpenShift CP4I Traitement des incidents liés à IBM MQ Operator

Si vous rencontrez des problèmes liés à IBM MQ Operator, appliquez les techniques de ce document pour les diagnostiquer et les résoudre.

Procédure

- «Identification et résolution des problèmes: accès aux données du gestionnaire de files d'attente», à la page 130

OpenShift CP4I Identification et résolution des problèmes: accès aux données du gestionnaire de files d'attente

Utilisez l'outil d'inspection de réservation de volume persistant pour accéder aux fichiers d'une réservation de volume persistant de gestionnaire de files d'attente dans laquelle un shell distant ne peut pas être établi sur le pod du gestionnaire de files d'attente. Cela peut être dû au fait que le pod est à l'état **`Error`** ou **`CrashLoopBackOff`** . Cet outil est conçu pour être utilisé avec les gestionnaires de files d'attente déployés par IBM MQ Operator.

Avant de commencer

Pour utiliser l'outil d'inspecteur PVC. vous devez avoir accès à l'espace de nom de votre gestionnaire de files d'attente.

Pourquoi et quand exécuter cette tâche

Pour faciliter le traitement des incidents, vous pouvez accéder aux données stockées sur les réservations de volume persistant (PVC) associées à un gestionnaire de files d'attente donné. Pour ce faire, vous utilisez un outil pour monter les réservations de volume persistant sur un ensemble de pods inspector. Vous pouvez ensuite obtenir un shell distant dans n'importe lequel des pods inspector pour lire les fichiers.

Selon le type de déploiement, entre un et trois pods inspector sont créés. Les volumes spécifiques à un pod donné d'un gestionnaire de files d'attente Native-HA ou Multi-Instance sont disponibles sur le pod de l'inspecteur PVC associé. Les volumes partagés sont disponibles sur tous les inspecteurs. Le nom du pod inspector contient le nom du pod de gestionnaire de files d'attente associé.

Procédure

1. Téléchargez l'outil MQ PVC inspector.

   L'outil est disponible ici: https://github.com/ibm-messaging/mq-pvc-tool.
2. Vérifiez que vous êtes connecté à votre cluster.
3. Recherchez le nom du gestionnaire de files d'attente et l'espace de nom dans lequel le gestionnaire de files d'attente s'exécute.
4. Exécutez l'outil inspector sur votre gestionnaire de files d'attente.
   a) Exécutez la commande suivante en spécifiant le nom de votre gestionnaire de files d'attente et son nom d'espace de nom.

   ```
   ./pvc-tool.sh queue_manager_name queue_manager_namespace_name
   ```

   b) Une fois l'outil terminé, exécutez la commande suivante pour afficher les pods d'inspecteur en cours de création.

   ```
   oc get pods
   ```

5. Affichez les fichiers montés sur le pod inspector.
   a) Chaque pod d'inspecteur de PVC est associé à un pod de gestionnaire de files d'attente, de sorte qu'il peut y avoir plusieurs pods d'inspecteur. Accédez à l'un de ces pods en exécutant la commande suivante:

      ```
      oc rsh pvc-inspector-pod-name
      ```

      Vous êtes placé dans le répertoire contenant les répertoires PVC montés.
   b) Ouvrez un interpréteur de commandes distant dans le pod, en exécutant la commande suivante:

      ```
      ls
      ```

   c) Vous pouvez voir les répertoires portant le même nom que les réservations de volume persistant qui ont été montées. Accédez aux fichiers des réservations de volume persistant du gestionnaire de files d'attente en parcourant ces répertoires. Pour afficher la liste des réservations de volume persistant, exécutez la commande suivante en dehors de la session shell distante:

      ```
      oc get pvc
      ```

   d) Nettoyez les pods créés par l'outil en exécutant la commande suivante:

      ```
      'oc delete pods -l tool=mq-pvc-inspector
      ```

OpenShift CP4I Référence d'API pour IBM MQ Operator

IBM MQ fournit un opérateur Kubernetes, qui fournit une intégration native avec la plateforme de conteneurs Red Hat OpenShift.

OpenShift CP4I Référence d'API pour mq.ibm.com/v1beta1

Vous pouvez utiliser l'API v1beta1 pour créer et gérer des ressources QueueManager.

OpenShift CP4I CD EUS *Référence relative à l'octroi de licence pour mq.ibm.com/v1beta1*

Versions de licence actuelles

La zone `spec.license.license` doit contenir l'identificateur de licence de la licence que vous acceptez. Les valeurs valides sont les suivantes :

Valeur de `spec.license.license`	Valeur de `spec.license.use`	Informations sur la licence	Versions IBM MQ applicables
`L-RJON-C7QG3S`	`Production` ou `NonProduction`	IBM Cloud Pak for Integration 2021.4.1	9.2.4 ou 9.2.5
`L-RJON-C7QFZX`	`Production` ou `NonProduction`	IBM Cloud Pak for Integration Limited Edition 2021.4.1	9.2.4 ou 9.2.5
`L-RJON-C5CSNH`	`Production` ou `NonProduction`	IBM Cloud Pak for Integration 2021.3.1	9.2.3 ou 9.2.4
`L-RJON-C5CSM2`	`Production` ou `NonProduction`	IBM Cloud Pak for Integration Limited Edition 2021.3.1	9.2.3 ou 9.2.4
`L-RJON-BZFQU2`	`Production` ou `NonProduction`	IBM Cloud Pak for Integration 2021.2.1	9.2.3

Valeur de **spec.license.license**	Valeur de **spec.license.use**	Informations sur la licence	Versions IBM MQ applicables
L-RJON-BZFQSB	Production ou NonProduction	IBM Cloud Pak for Integration Edition limitée 2021.2.1	9.2.3
L-RJON-BUVMQX	Production ou NonProduction	IBM Cloud Pak for Integration 2020.4.1	9.2.0 EUS ou 9.2.1
L-RJON-BUVMYB	Production ou NonProduction	IBM Cloud Pak for Integration Edition limitée 2020.4.1	9.2.0 EUS ou 9.2.1
L-APIG-BZDDDY	Production	IBM MQ Advanced et IBM MQ Advanced pour l'environnement de non-production 9.2 -07/2021	9.2.3, 9.2.4 ou 9.2.5
L-APIG-BYHCL7	Development	IBM MQ Advanced for Developers (non garantie) V9.2 -07/2021	9.2.3, 9.2.4 ou 9.2.5
L-APIG-BVJJB3	Production	IBM MQ Advanced et IBM MQ Advanced for Non-Production Environment 9.2 - 03/2021	9.2.2
L-APIG-BMJJBM	Production	IBM MQ Advanced V9.2	9.2.0 CD ou 9.2.1
L-APIG-BMKG5H	Development	IBM MQ Advanced for Developers (non garantie) V9.2	9.2.0 CD, 9.2.1 ou 9.2.2

Notez que la *version* de licence est spécifiée, et qu'elle n'est pas toujours identique à la version d'IBM MQ.

Versions de licence plus anciennes

La zone `spec.license.license` doit contenir l'identificateur de licence de la licence que vous acceptez. Les valeurs valides sont les suivantes :

Valeur de **spec.license.license**	Valeur de **spec.license.use**	Informations sur la licence	Versions IBM MQ applicables
L-RJON-BXUPZ2	Production ou NonProduction	IBM Cloud Pak for Integration 2021.1.1	9.2.2
L-RJON-BXUQ34	Production ou NonProduction	IBM Cloud Pak for Integration Edition limitée 2021.1.1	9.2.2
L-RJON-BYRMYW	NonProduction	IBM Cloud Pak for Integration Eval-Demo 2021.1.1. Edition anticipée pour une utilisation avec Native HA avec IBM MQ Operator 1.5 uniquement.	9.2.2
L-RJON-BQPGWD	Production ou NonProduction	IBM Cloud Pak for Integration 2020.3.1	9.2.0 CD
L-RJON-BN7PN3	Production ou NonProduction	IBM Cloud Pak for Integration 2020.2.1	9.1.5 ou 9.2.0 CD
L-RJON-BPHL2Y	Production ou NonProduction	IBM Cloud Pak for Integration Edition limitée 2020.2.1	9.1.5
L-APIG-BJAKBF	Production	IBM MQ Advanced V9.1 -04/2020	9.1.5
L-APIG-BM7GDH	Development	IBM MQ Advanced for Developers (non garantie) V9.1 -04/2020	9.1.5

Notez que la *version* de licence est spécifiée, et qu'elle n'est pas toujours identique à la version d'IBM MQ.

OpenShift CP4I *Référence d'API pour le gestionnaire de files d'attente (mq.ibm.com/v1beta1)*

Gestionnaire de files d'attente

Un gestionnaire de files d'attente est un serveur IBM MQ qui fournit des services de mise en file d'attente et de publication/abonnement à des applications.

Zone	Description
Chaîne `apiVersion`	APIVersion définit le schéma versionné de cette représentation d'un objet. Les serveurs doivent convertir les schémas reconnus dans la valeur interne la plus récente et peuvent rejeter les valeurs non reconnues. Pour plus d'informations : https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources.
Chaîne `kind`	Kind est une valeur de chaîne qui représente la ressource REST représentée par cet objet. Les serveurs peuvent déduire cette valeur du noeud final auquel le client soumet les demandes. Elle ne peut pas être mise à jour. Casse mixte. Pour plus d'informations : https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds.
`metadata`	
`spec` QueueManagerSpec	Etat souhaité pour le gestionnaire de files d'attente.
`status` QueueManagerStatus	Etat observé pour le gestionnaire de files d'attente.

.spec

Etat souhaité pour le gestionnaire de files d'attente.

Apparaît dans :

- «Gestionnaire de files d'attente», à la page 133

Zone	Description
`affinity`	Règles d'affinité Kubernetes standard. Pour plus d'informations, voir https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.17/#affinity-v1-core.
`annotations` Annotations	La zone annotations sert de passe-système pour les annotations de pod. Les utilisateurs peuvent ajouter n'importe quelle annotation dans cette zone et l'appliquer au pod. Les annotations ici écrasent les annotations par défaut si elles sont fournies. Requiert MQ Operator 1.3.0 ou version ultérieure.
Tableau `imagePullSecrets` LocalObjectReference	Liste facultative de références à des secrets dans le même espace de nom, à utiliser pour extraire les images utilisées par ce gestionnaire de files d'attente. Si cette liste est spécifiée, les secrets sont transmis à des implémentations de programme d'extraction individuelles pour que celles-ci puissent les utiliser. Par exemple, dans le cas de docker, seuls les secrets de type DockerConfig sont honorés. Pour plus d'informations, voir https://kubernetes.io/docs/concepts/containers/images#specifying-imagepullsecrets-on-a-pod.
`labels` Libellés	La zone labels sert de passe-système pour les libellés de pod. Les utilisateurs peuvent ajouter n'importe quel libellé dans cette zone et l'appliquer au pod. Les libellés ici remplacent les libellés par défaut s'ils sont fournis. Requiert MQ Operator 1.3.0 ou version ultérieure.

Zone	Description
`license` License	Paramètres contrôlant votre acceptation de la licence et les métriques de licence à utiliser.
`pki` PKI	Paramètres de l'infrastructure à clés publiques (PKI) pour la définition de clés et de certificats à utiliser avec Transport Layer Security (TLS) ou MQ Advanced Message Security (AMS).
`queueManager` QueueManagerConfig	Paramètres pour le conteneur du gestionnaire de files d'attente et le gestionnaire de files d'attente sous-jacent.
`securityContext` SecurityContext	Paramètres de sécurité à ajouter au pod securityContext du gestionnaire de files d'attente.
`template` Template	Création avancée de modèle pour les ressources Kubernetes. Le modèle permet aux utilisateurs d'indiquer comment IBM MQ génère les ressources Kubernetes sous-jacentes, telles que les objets StatefulSet, Pods et Services. Ce paramètre est réservé aux utilisateurs avancés, car il peut interrompre le fonctionnement normal de MQ s'il n'est pas utilisé correctement. Toute valeur spécifiée ailleurs dans la ressource QueueManager sera remplacée par les paramètres figurant dans le modèle.
Entier `terminationGracePeriodSeconds`	Durée facultative en secondes au bout de laquelle le pod doit s'arrêter correctement. La valeur doit être un entier non négatif. La valeur zéro indique la suppression immédiate. Heure cible à laquelle l'arrêt du gestionnaire de files d'attente est tenté, avec l'escalade des phases de la déconnexion des applications. Les tâches de maintenance essentielles du gestionnaire de files d'attente sont interrompues si nécessaire. La valeur par défaut est 30 secondes.
`tracing` TracingConfig	Paramètres de traçage de l'intégration au tableau de bord des opérations de Cloud Pak for Integration.
Chaîne `version`	Paramètre qui contrôle la version de MQ qui sera utilisée (requis). Exemple : `9.1.5.0-r2` spécifie la version 9.1.5.0 de MQ qui utilise la deuxième révision de l'image de conteneur. Les correctifs propres au conteneur sont souvent appliqués dans des révisions, comme les correctifs de l'image de base.
`web` WebServerConfig	Paramètres pour le serveur Web MQ.

.spec.annotations

La zone annotations sert de passe-système pour les annotations de pod. Les utilisateurs peuvent ajouter n'importe quelle annotation dans cette zone et l'appliquer au pod. Les annotations ici écrasent les annotations par défaut si elles sont fournies. Requiert MQ Operator 1.3.0 ou version ultérieure.

Apparaît dans :

- «.spec», à la page 133

.spec.imagePullSecrets

LocalObjectReference contient suffisamment d'informations pour vous permettre de localiser l'objet référencé dans le même espace de nom.

Apparaît dans :

- «.spec», à la page 133

Zone	Description
Chaîne `name`	Nom du référent. Pour plus d'informations: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names À FAIRE : ajoutez d'autres champs utiles : apiVersion, kind, uid?.

.spec.labels

La zone labels sert de passe-système pour les libellés de pod. Les utilisateurs peuvent ajouter n'importe quel libellé dans cette zone et l'appliquer au pod. Les libellés ici remplacent les libellés par défaut s'ils sont fournis. Requiert MQ Operator 1.3.0 ou version ultérieure.

Apparaît dans :

- «.spec», à la page 133

.spec.license

Paramètres contrôlant votre acceptation de la licence et les métriques de licence à utiliser.

Apparaît dans :

- «.spec», à la page 133

Zone	Description
Valeur booléenne `accept`	Indique si vous acceptez ou non la licence associée à ce logiciel (requis).
Chaîne `license`	Identificateur de la licence que vous acceptez. Il doit s'agir de l'identificateur de licence correct pour la version de MQ que vous utilisez. Voir http://ibm.biz/BdqvCF pour les valeurs valides.
Chaîne `metric`	Paramètre qui spécifie la métrique de licence à utiliser. Par exemple, `ProcessorValueUnit`, `VirtualProcessorCore` ou `ManagedVirtualServer`. Prend par défaut la valeur `ProcessorValueUnit` lors de l'utilisation d'une licence MQ et de `VirtualProcessorCore` lors de l'utilisation d'une licence Cloud Pak for Integration.
Chaîne `use`	Paramètre qui contrôle la façon dont le logiciel sera utilisé, où la licence prend en charge plusieurs utilisations. Voir http://ibm.biz/BdqvCF pour les valeurs valides.

.spec.pki

Paramètres de l'infrastructure à clés publiques (PKI) pour la définition de clés et de certificats à utiliser avec Transport Layer Security (TLS) ou MQ Advanced Message Security (AMS).

Apparaît dans :

- «.spec», à la page 133

Zone	Description
Tableau `keys` PKISource	Clés privées à ajouter au référentiel de clés du gestionnaire de files d'attente.
Tableau `trust` PKISource	Certificats à ajouter au référentiel de clés du gestionnaire de files d'attente.

.spec.pki.keys

PKISource définit une source des informations de l'infrastructure à clés publiques (PKI), comme des clés ou des certificats.

Apparaît dans :

- «.spec.pki», à la page 135

Zone	Description
Chaîne `name`	Name est utilisé comme libellé pour la clé ou le certificat. Il doit s'agir d'une chaîne alphanumérique en minuscules.
`secret` Secret	Fournissez une clé à l'aide d'un secret Kubernetes.

.spec.pki.keys.secret

Fournissez une clé à l'aide d'un secret Kubernetes.

Apparaît dans :

- «.spec.pki.keys», à la page 135

Zone	Description
Tableau `items`	Clés dans le secret Kubernetes qui doivent être ajoutées au conteneur du gestionnaire de files d'attente.
Chaîne `secretName`	Nom du secret Kubernetes.

.spec.pki.trust

PKISource définit une source des informations de l'infrastructure à clés publiques (PKI), comme des clés ou des certificats.

Apparaît dans :

- «.spec.pki», à la page 135

Zone	Description
Chaîne `name`	Name est utilisé comme libellé pour la clé ou le certificat. Il doit s'agir d'une chaîne alphanumérique en minuscules.
`secret` Secret	Fournissez une clé à l'aide d'un secret Kubernetes.

.spec.pki.trust.secret

Fournissez une clé à l'aide d'un secret Kubernetes.

Apparaît dans :

- «.spec.pki.trust», à la page 136

Zone	Description
Tableau `items`	Clés dans le secret Kubernetes qui doivent être ajoutées au conteneur du gestionnaire de files d'attente.
Chaîne `secretName`	Nom du secret Kubernetes.

.spec.queueManager

Paramètres pour le conteneur du gestionnaire de files d'attente et le gestionnaire de files d'attente sous-jacent.

Apparaît dans :

- «.spec», à la page 133

Zone	Description
`availability` Availability	Paramètres de disponibilité pour le gestionnaire de files d'attente, indiquant par exemple si une paire actif-secours ou Native HA doit être utilisée.
Valeur booléenne `debug`	Indique si les messages de débogage du code propre au conteneur doivent être consignés ou non dans le journal du conteneur. La valeur par défaut est false.
Chaîne `image`	Image de conteneur à utiliser.
Chaîne `imagePullPolicy`	Paramètre qui contrôle à quel moment le kubelet tente d'extraire l'image spécifiée. La valeur par défaut est `IfNotPresent`.
Tableau `ini` INISource	Paramètres permettant de fournir les informations INI pour le gestionnaire de files d'attente. Requiert MQ Operator 1.1.0 ou version ultérieure.
`livenessProbe` QueueManagerLivenessProbe	Paramètres qui contrôlent la sonde de non-défaillance.
Chaîne `logFormat`	Indique le format de journal à utiliser pour ce conteneur. Utilisez `JSON` pour les journaux au format JSON du conteneur. Utilisez `Basic` pour les messages au format texte. La valeur par défaut est `Basic`.
`metrics` QueueManagerMetrics	Paramètres pour les métriques de style Prometheus.
Tableau `mqsc` MQSCSource	Paramètres permettant de fournir des informations MQSC pour le gestionnaire de files d'attente. Requiert MQ Operator 1.1.0 ou version ultérieure.
Chaîne `name`	Nom du gestionnaire de files d'attente MQ sous-jacent, s'il est différent de metadata.name. Servez-vous de cette zone si vous voulez utiliser un nom de gestionnaire de files d'attente qui n'est pas conforme aux règles Kubernetes relatives aux noms (par exemple, un nom incluant des majuscules).
`readinessProbe` QueueManagerReadinessProbe	Paramètres qui contrôlent la sonde de vigilance.
`resources` Resources	Paramètres qui contrôlent les exigences relatives aux ressources.
`route` Route	Paramètres de la route du gestionnaire de files d'attente. Requiert MQ Operator 1.4.0 ou version ultérieure.
`startupProbe` StartupProbe	Paramètres qui contrôlent la sonde de démarrage. Ne s'applique qu'aux déploiements MultiInstance et NativeHA. Requiert MQ Operator 1.5.0 ou une version ultérieure.
`storage` QueueManagerStorage	Paramètres de stockage pour contrôler l'utilisation des volumes persistants et des classes de stockage par le gestionnaire de files d'attente.

.spec.queueManager.availability

Paramètres de disponibilité pour le gestionnaire de files d'attente, indiquant par exemple si une paire actif-secours ou Native HA doit être utilisée.

Apparaît dans :

- «.spec.queueManager», à la page 136

Zone	Description
`tls` Tls	Paramètres TLS facultatifs permettant de configurer les communications sécurisées entre les répliques NativeHA. Requiert MQ Operator 1.5.0 ou une version ultérieure.

Zone	Description
Chaîne `type`	Type de disponibilité à utiliser. Utilisez `SingleInstance` pour un pod unique, qui sera redémarré automatiquement (dans certains cas) par Kubernetes. Utilisez `MultiInstance` pour une paire de pods, dont l'un est le gestionnaire de files d'attente `actif`, et l'autre un gestionnaire de secours. Utilisez `NativeHA` pour la réplication Native HA (requiert MQ Operator 1.5.0 ou une version ultérieure). La valeur par défaut est `SingleInstance`. Pour plus d'informations, voir http://ibm.biz/BdqAQa.
Chaîne `updateStrategy`	Stratégie de mise à jour à utiliser pour les gestionnaires de files d'attente MultiInstance et NativeHA. Utilisez `RollingUpdate` pour activer les mises à jour automatiques chaque fois que la configuration du gestionnaire de files d'attente change. Utilisez `OnDelete` pour désactiver les mises à jour automatiques. Les modifications du gestionnaire de files d'attente ne seront appliquées que lorsque les pods sont supprimés (y compris les suppressions de pods déclenchées par des facteurs externes). La valeur par défaut est `RollingUpdate`. Nécessite MQ Operator 1.6.0 ou version supérieure.

.spec.queueManager.availability.tls

Paramètres TLS facultatifs permettant de configurer les communications sécurisées entre les répliques NativeHA. Requiert MQ Operator 1.5.0 ou une version ultérieure.

Apparaît dans :

- «.spec.queueManager.availability», à la page 137

Zone	Description
Chaîne `cipherSpec`	Nom du CipherSpec pour une connexion TLS NativeHA.
Chaîne `secretName`	Nom du secret Kubernetes.

.spec.queueManager.ini

Source des fichiers de configuration INI.

Apparaît dans :

- «.spec.queueManager», à la page 136

Zone	Description
`configMap` ConfigMapINISource	ConfigMap représente une mappe de configuration Kubernetes qui contient des informations INI.
`secret` SecretINISource	Secret représente un secret Kubernetes qui contient des informations INI.

.spec.queueManager.ini.configMap

ConfigMap représente une mappe de configuration Kubernetes qui contient des informations INI.

Apparaît dans :

- «.spec.queueManager.ini», à la page 138

Zone	Description
Tableau `items`	Clés dans la source Kubernetes, qui doivent être appliquées.
Chaîne `name`	Nom de la source Kubernetes.

.spec.queueManager.ini.secret

Secret représente un secret Kubernetes qui contient des informations INI.

Apparaît dans :

- «.spec.queueManager.ini», à la page 138

Zone	Description
Tableau `items`	Clés dans la source Kubernetes, qui doivent être appliquées.
Chaîne `name`	Nom de la source Kubernetes.

.spec.queueManager.livenessProbe

Paramètres qui contrôlent la sonde de non-défaillance.

Apparaît dans :

- «.spec.queueManager», à la page 136

Zone	Description
Entier `failureThreshold`	Nombre minimal d'échecs consécutifs nécessaire pour que l'on considère que la sonde a échoué après une réussite. La valeur par défaut est 1.
Entier `initialDelaySeconds`	Nombre de secondes après le démarrage du conteneur avant que la sonde ne soit initiée. La valeur par défaut est 90 secondes pour SingleInstance. La valeur par défaut est 0 seconde pour les déploiements MultiInstance et NativeHA. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
Entier `periodSeconds`	Fréquence d'exécution de la sonde (en secondes). La valeur par défaut est 10 secondes.
Entier `successThreshold`	Nombre minimal de réussites consécutives nécessaire pour que l'on considère que la sonde a abouti après un échec. La valeur par défaut est 1.
Entier `timeoutSeconds`	Délai d'expiration de la sonde, en secondes. La valeur par défaut est 5 secondes. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.queueManager.metrics

Paramètres pour les métriques de style Prometheus.

Apparaît dans :

- «.spec.queueManager», à la page 136

Zone	Description
Valeur booléenne `enabled`	Indique si un noeud final doit être activé pour permettre à Prometheus de collecter des métriques compatibles avec Prometheus. La valeur par défaut est true.

.spec.queueManager.mqsc

Source des fichiers de configuration MQSC.

Apparaît dans :

- «.spec.queueManager», à la page 136

Zone	Description
`configMap` ConfigMapMQSCSource	ConfigMap représente une mappe de configuration Kubernetes qui contient des informations MQSC.
`secret` SecretMQSCSource	Secret représente un Secret Kubernetes qui contient des informations MQSC.

.spec.queueManager.mqsc.configMap

ConfigMap représente une mappe de configuration Kubernetes qui contient des informations MQSC.

Apparaît dans :

- «.spec.queueManager.mqsc», à la page 139

Zone	Description
Tableau `items`	Clés dans la source Kubernetes, qui doivent être appliquées.
Chaîne `name`	Nom de la source Kubernetes.

.spec.queueManager.mqsc.secret

Secret représente un Secret Kubernetes qui contient des informations MQSC.

Apparaît dans :

- «.spec.queueManager.mqsc», à la page 139

Zone	Description
Tableau `items`	Clés dans la source Kubernetes, qui doivent être appliquées.
Chaîne `name`	Nom de la source Kubernetes.

.spec.queueManager.readinessProbe

Paramètres qui contrôlent la sonde de vigilance.

Apparaît dans :

- «.spec.queueManager», à la page 136

Zone	Description
Entier `failureThreshold`	Nombre minimal d'échecs consécutifs nécessaire pour que l'on considère que la sonde a échoué après une réussite. La valeur par défaut est 1.
Entier `initialDelaySeconds`	Nombre de secondes après le démarrage du conteneur avant que la sonde ne soit initiée. La valeur par défaut est 10 secondes pour SingleInstance. La valeur par défaut est 0 pour les déploiements MultiInstance et NativeHA. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
Entier `periodSeconds`	Fréquence d'exécution de la sonde (en secondes). La valeur par défaut est 5 secondes.
Entier `successThreshold`	Nombre minimal de réussites consécutives nécessaire pour que l'on considère que la sonde a abouti après un échec. La valeur par défaut est 1.
Entier `timeoutSeconds`	Délai d'expiration de la sonde, en secondes. La valeur par défaut est 3 secondes. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.queueManager.resources

Paramètres qui contrôlent les exigences relatives aux ressources.

Apparaît dans :

- «.spec.queueManager», à la page 136

Zone	Description
`limits` Limits	Paramètres d'unité centrale et de mémoire.
`requests` Requests	Paramètres d'unité centrale et de mémoire.

.spec.queueManager.resources.limits

Paramètres d'unité centrale et de mémoire.

Apparaît dans :

- «.spec.queueManager.resources», à la page 141

Zone	Description
`cpu`	
`memory`	

.spec.queueManager.resources.requests

Paramètres d'unité centrale et de mémoire.

Apparaît dans :

- «.spec.queueManager.resources», à la page 141

Zone	Description
`cpu`	
`memory`	

.spec.queueManager.route

Paramètres de la route du gestionnaire de files d'attente. Requiert MQ Operator 1.4.0 ou version ultérieure.

Apparaît dans :

- «.spec.queueManager», à la page 136

Zone	Description
Valeur booléenne `enabled`	Indique si la route doit être activée ou non. La valeur par défaut est true.

.spec.queueManager.startupProbe

Paramètres qui contrôlent la sonde de démarrage. Ne s'applique qu'aux déploiements MultiInstance et NativeHA. Requiert MQ Operator 1.5.0 ou une version ultérieure.

Apparaît dans :

- «.spec.queueManager», à la page 136

Zone	Description
Entier `failureThreshold`	Nombre minimal d'échecs consécutifs nécessaire pour considérer que la sonde a échoué. La valeur par défaut est 60.
Entier `initialDelaySeconds`	Nombre de secondes après le démarrage du conteneur avant que la sonde ne soit initiée. La valeur par défaut est 0 seconde. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
Entier `periodSeconds`	Fréquence d'exécution de la sonde (en secondes). La valeur par défaut est 5 secondes.
Entier `successThreshold`	Nombre minimal de réussites consécutives nécessaire pour considérer que la sonde a réussi. La valeur par défaut est 1.
Entier `timeoutSeconds`	Délai d'expiration de la sonde, en secondes. La valeur par défaut est 5 secondes. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.queueManager.storage

Paramètres de stockage pour contrôler l'utilisation des volumes persistants et des classes de stockage par le gestionnaire de files d'attente.

Apparaît dans :

- «.spec.queueManager», à la page 136

Zone	Description
Chaîne `defaultClass`	Classe de stockage à appliquer à tous les volumes persistants de ce gestionnaire de files d'attente par défaut. Les volumes persistants spécifiques peuvent définir leur propre classe de stockage qui remplacera ce paramètre de classe de stockage par défaut. Si `type of availability` a pour valeur `SingleInstance` ou `NativeHA`, la classe de stockage peut être de type ReadWriteOnce ou ReadWriteMany. Si `type of availability` est `MultiInstance`, la classe de stockage doit être de type ReadWriteMany.
Booléen `defaultDeleteClaim`	Indique si tous les volumes doivent être supprimés lorsque le gestionnaire de files d'attente est supprimé. Les volumes persistants spécifiques peuvent définir leur propre valeur pour deleteClaim qui remplacera ce paramètre defaultDeleteClaim. La valeur par défaut est false.
`persistedData` QueueManagerOptionalVolume	Détails du volume persistant pour les données conservées par MQ, notamment la configuration, les files d'attente et les messages. Requis en cas d'utilisation d'un gestionnaire de files d'attente multi-instance.
`queueManager` QueueManagerVolume	Volume persistant par défaut pour les données normalement sous `/var/mqm`. Il contient toutes les données conservées et tous les journaux de reprise, si aucun autre volume n'est spécifié.
`recoveryLogs` QueueManagerOptionalVolume	Détails du volume persistant pour les journaux de reprise MQ. Requis en cas d'utilisation d'un gestionnaire de files d'attente multi-instance.

.spec.queueManager.storage.persistedData

Détails du volume persistant pour les données conservées par MQ, notamment la configuration, les files d'attente et les messages. Requis en cas d'utilisation d'un gestionnaire de files d'attente multi-instance.

Apparaît dans :

- «.spec.queueManager.storage», à la page 142

Zone	Description
Chaîne `class`	Classe d'archivage à utiliser pour ce volume. Valide uniquement si `type` a pour valeur `persistent-claim`. Si `type of availability` a pour valeur `SingleInstance` ou `NativeHA`, la classe de stockage peut être de type ReadWriteOnce ou ReadWriteMany. Si `type of availability` est `MultiInstance`, la classe de stockage doit être de type ReadWriteMany.
Booléen `deleteClaim`	Indique si ce volume doit être supprimé lorsque le gestionnaire de files d'attente est supprimé.
Valeur booléenne `enabled`	Indique si ce volume doit être activé en tant que volume distinct ou placé dans le volume `queueManager` par défaut, ou non. La valeur par défaut est false.
Chaîne `size`	Taille du volume persistant à transmettre à Kubernetes, incluant les unités SI. Valide uniquement si `type` a pour valeur `persistent-claim`. Exemple : `2Gi`. La valeur par défaut est `2Gi`.
Chaîne `sizeLimit`	Limite de taille en cas d'utilisation d'un volume de type `ephemeral`. Les fichiers continuent d'être écrits dans un répertoire temporaire ; ainsi, vous pouvez utiliser cette option pour limiter la taille. Valide uniquement si `type` a pour valeur `ephemeral`.
Chaîne `type`	Type de volume à utiliser. Choisissez `ephemeral` pour utiliser un stockage non persistant ou `persistent-claim` pour utiliser un volume persistant. La valeur par défaut est `persistent-claim`.

.spec.queueManager.storage.queueManager

Volume persistant par défaut pour les données normalement sous `/var/mqm`. Il contient toutes les données conservées et tous les journaux de reprise, si aucun autre volume n'est spécifié.

Apparaît dans :

- «.spec.queueManager.storage», à la page 142

Zone	Description
Chaîne `class`	Classe d'archivage à utiliser pour ce volume. Valide uniquement si `type` a pour valeur `persistent-claim`. Si `type of availability` a pour valeur `SingleInstance` ou `NativeHA`, la classe de stockage peut être de type ReadWriteOnce ou ReadWriteMany. Si `type of availability` est `MultiInstance`, la classe de stockage doit être de type ReadWriteMany.
Booléen `deleteClaim`	Indique si ce volume doit être supprimé lorsque le gestionnaire de files d'attente est supprimé.
Chaîne `size`	Taille du volume persistant à transmettre à Kubernetes, incluant les unités SI. Valide uniquement si `type` a pour valeur `persistent-claim`. Exemple : `2Gi`. La valeur par défaut est `2Gi`.
Chaîne `sizeLimit`	Limite de taille en cas d'utilisation d'un volume de type `ephemeral`. Les fichiers continuent d'être écrits dans un répertoire temporaire ; ainsi, vous pouvez utiliser cette option pour limiter la taille. Valide uniquement si `type` a pour valeur `ephemeral`.
Chaîne `type`	Type de volume à utiliser. Choisissez `ephemeral` pour utiliser un stockage non persistant ou `persistent-claim` pour utiliser un volume persistant. La valeur par défaut est `persistent-claim`.

.spec.queueManager.storage.recoveryLogs

Détails du volume persistant pour les journaux de reprise MQ. Requis en cas d'utilisation d'un gestionnaire de files d'attente multi-instance.

Apparaît dans :

- «.spec.queueManager.storage», à la page 142

Zone	Description
Chaîne `class`	Classe d'archivage à utiliser pour ce volume. Valide uniquement si `type` a pour valeur `persistent-claim`. Si `type of availability` a pour valeur `SingleInstance` ou `NativeHA`, la classe de stockage peut être de type ReadWriteOnce ou ReadWriteMany. Si `type of availability` est `MultiInstance`, la classe de stockage doit être de type ReadWriteMany.
Booléen `deleteClaim`	Indique si ce volume doit être supprimé lorsque le gestionnaire de files d'attente est supprimé.
Valeur booléenne `enabled`	Indique si ce volume doit être activé en tant que volume distinct ou placé dans le volume `queueManager` par défaut, ou non. La valeur par défaut est false.
Chaîne `size`	Taille du volume persistant à transmettre à Kubernetes, incluant les unités SI. Valide uniquement si `type` a pour valeur `persistent-claim`. Exemple : `2Gi`. La valeur par défaut est `2Gi`.
Chaîne `sizeLimit`	Limite de taille en cas d'utilisation d'un volume de type `ephemeral`. Les fichiers continuent d'être écrits dans un répertoire temporaire ; ainsi, vous pouvez utiliser cette option pour limiter la taille. Valide uniquement si `type` a pour valeur `ephemeral`.
Chaîne `type`	Type de volume à utiliser. Choisissez `ephemeral` pour utiliser un stockage non persistant ou `persistent-claim` pour utiliser un volume persistant. La valeur par défaut est `persistent-claim`.

.spec.securityContext

Paramètres de sécurité à ajouter au pod securityContext du gestionnaire de files d'attente.

Apparaît dans :

- «.spec», à la page 133

Zone	Description
Entier `fsGroup`	Groupe supplémentaire spécial qui s'applique à tous les conteneurs dans un pod. Certains types de volume permettent à Kubelet de changer la propriété de ce volume pour être la propriété de la nacelle : 1. Le GID propriétaire sera le FSGroup 2. Le bit setgid est défini (les nouveaux fichiers créés dans le volume seront la propriété de FSGroup) 3. Les bits d'autorisation sont sujets d'un OR avec rw-rw ---- Si la configuration est annulée, Kubelet ne modifiera pas la propriété et les autorisations de n'importe quel volume.

Zone	Description
Valeur booléenne `initVolumeAsRoot`	Elle a un impact sur le contexte de sécurité utilisé par le conteneur qui initialise le volume persistant. Définissez la valeur `true` si vous utilisez un fournisseur de stockage qui exige que vous soyez l'utilisateur racine pour pouvoir accéder aux volumes nouvellement mis à disposition. La valeur `true` a un impact sur l'objet Contraintes de contexte de sécurité (SCC) que vous pouvez utiliser, et le démarrage du gestionnaire de files d'attente peut échouer si vous n'êtes pas autorisé à utiliser un objet SCC autorisant l'utilisateur racine. La valeur par défaut est false. Pour plus d'informations, voir https://docs.openshift.com/container-platform/latest/authentication/managing-security-context-constraints.html.
Tableau `supplementalGroups`	Une liste des groupes appliqués au premier processus s'exécute dans chaque conteneur, en plus du GID principal du conteneur. Si cette liste n'est pas spécifiée, aucun groupe n'est ajouté à aucun conteneur.

.spec.template

Création avancée de modèle pour les ressources Kubernetes. Le modèle permet aux utilisateurs d'indiquer comment IBM MQ génère les ressources Kubernetes sous-jacentes, telles que les objets StatefulSet, Pods et Services. Ce paramètre est réservé aux utilisateurs avancés, car il peut interrompre le fonctionnement normal de MQ s'il n'est pas utilisé correctement. Toute valeur spécifiée ailleurs dans la ressource QueueManager sera remplacée par les paramètres figurant dans le modèle.

Apparaît dans :

- «.spec», à la page 133

Zone	Description
pod	Substitutions pour le modèle utilisé pour le pod. Voir https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.17/#podspec-v1-core.

.spec.tracing

Paramètres de traçage de l'intégration au tableau de bord des opérations de Cloud Pak for Integration.

Apparaît dans :

- «.spec», à la page 133

Zone	Description
`agent` TracingAgent	Dans Cloud Pak for Integration uniquement, vous pouvez configurer des paramètres pour l'agent de trace facultatif.
`collector` TracingCollector	Dans Cloud Pak for Integration uniquement, vous pouvez configurer des paramètres pour le collecteur de trace facultatif.
Valeur booléenne `enabled`	Indique si l'intégration au tableau de bord des opérations de Cloud Pak for Integration doit être activée ou non, via la fonction de trace. La valeur par défaut est false.
Chaîne `namespace`	Espace de nom dans lequel le tableau de bord des opérations de Cloud Pak for Integration est installé.

.spec.tracing.agent

Dans Cloud Pak for Integration uniquement, vous pouvez configurer des paramètres pour l'agent de trace facultatif.

Apparaît dans :

- «.spec.tracing», à la page 145

Zone	Description
Chaîne `image`	Image de conteneur à utiliser.
Chaîne `imagePullPolicy`	Paramètre qui contrôle à quel moment le kubelet tente d'extraire l'image spécifiée. La valeur par défaut est `IfNotPresent`.
`livenessProbe` TracingProbe	Paramètres qui contrôlent la sonde de non-défaillance.
`readinessProbe` TracingProbe	Paramètres qui contrôlent la sonde de vigilance.

.spec.tracing.agent.livenessProbe

Paramètres qui contrôlent la sonde de non-défaillance.

Apparaît dans :

- «.spec.tracing.agent», à la page 145

Zone	Description
Entier `failureThreshold`	Nombre minimal d'échecs consécutifs nécessaire pour que l'on considère que la sonde a échoué après une réussite. La valeur par défaut est 1.
Entier `initialDelaySeconds`	Nombre de secondes après le démarrage du conteneur avant que des sondes de non-défaillance ne soient initiées. La valeur par défaut est 10 secondes. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
Entier `periodSeconds`	Fréquence d'exécution de la sonde (en secondes). La valeur par défaut est 10 secondes.
Entier `successThreshold`	Nombre minimal de réussites consécutives nécessaire pour que l'on considère que la sonde a abouti après un échec. La valeur par défaut est 1.
Entier `timeoutSeconds`	Délai d'expiration de la sonde, en secondes. La valeur par défaut est 2 secondes. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.tracing.agent.readinessProbe

Paramètres qui contrôlent la sonde de vigilance.

Apparaît dans :

- «.spec.tracing.agent», à la page 145

Zone	Description
Entier `failureThreshold`	Nombre minimal d'échecs consécutifs nécessaire pour que l'on considère que la sonde a échoué après une réussite. La valeur par défaut est 1.
Entier `initialDelaySeconds`	Nombre de secondes après le démarrage du conteneur avant que des sondes de non-défaillance ne soient initiées. La valeur par défaut est 10 secondes. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
Entier `periodSeconds`	Fréquence d'exécution de la sonde (en secondes). La valeur par défaut est 10 secondes.

Zone	Description
Entier `successThreshold`	Nombre minimal de réussites consécutives nécessaire pour que l'on considère que la sonde a abouti après un échec. La valeur par défaut est 1.
Entier `timeoutSeconds`	Délai d'expiration de la sonde, en secondes. La valeur par défaut est 2 secondes. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.tracing.collector

Dans Cloud Pak for Integration uniquement, vous pouvez configurer des paramètres pour le collecteur de trace facultatif.

Apparaît dans :

- «.spec.tracing», à la page 145

Zone	Description
Chaîne `image`	Image de conteneur à utiliser.
Chaîne `imagePullPolicy`	Paramètre qui contrôle à quel moment le kubelet tente d'extraire l'image spécifiée. La valeur par défaut est `IfNotPresent`.
`livenessProbe` TracingProbe	Paramètres qui contrôlent la sonde de non-défaillance.
`readinessProbe` TracingProbe	Paramètres qui contrôlent la sonde de vigilance.

.spec.tracing.collector.livenessProbe

Paramètres qui contrôlent la sonde de non-défaillance.

Apparaît dans :

- «.spec.tracing.collector», à la page 147

Zone	Description
Entier `failureThreshold`	Nombre minimal d'échecs consécutifs nécessaire pour que l'on considère que la sonde a échoué après une réussite. La valeur par défaut est 1.
Entier `initialDelaySeconds`	Nombre de secondes après le démarrage du conteneur avant que des sondes de non-défaillance ne soient initiées. La valeur par défaut est 10 secondes. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
Entier `periodSeconds`	Fréquence d'exécution de la sonde (en secondes). La valeur par défaut est 10 secondes.
Entier `successThreshold`	Nombre minimal de réussites consécutives nécessaire pour que l'on considère que la sonde a abouti après un échec. La valeur par défaut est 1.
Entier `timeoutSeconds`	Délai d'expiration de la sonde, en secondes. La valeur par défaut est 2 secondes. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.tracing.collector.readinessProbe

Paramètres qui contrôlent la sonde de vigilance.

Apparaît dans :

- «.spec.tracing.collector», à la page 147

Zone	Description
Entier `failureThreshold`	Nombre minimal d'échecs consécutifs nécessaire pour que l'on considère que la sonde a échoué après une réussite. La valeur par défaut est 1.
Entier `initialDelaySeconds`	Nombre de secondes après le démarrage du conteneur avant que des sondes de non-défaillance ne soient initiées. La valeur par défaut est 10 secondes. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
Entier `periodSeconds`	Fréquence d'exécution de la sonde (en secondes). La valeur par défaut est 10 secondes.
Entier `successThreshold`	Nombre minimal de réussites consécutives nécessaire pour que l'on considère que la sonde a abouti après un échec. La valeur par défaut est 1.
Entier `timeoutSeconds`	Délai d'expiration de la sonde, en secondes. La valeur par défaut est 2 secondes. Pour plus d'informations : https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.web

Paramètres pour le serveur Web MQ.

Apparaît dans :

- «.spec», à la page 133

Zone	Description
Valeur booléenne `enabled`	Indique si le serveur Web doit être activé ou non. La valeur par défaut est false.

.status

Etat observé pour le gestionnaire de files d'attente.

Apparaît dans :

- «Gestionnaire de files d'attente», à la page 133

Zone	Description
Chaîne `adminUiUrl`	URL de l'interface utilisateur d'administration.
`availability` Availability	Statut de disponibilité du gestionnaire de files d'attente.
Tableau `conditions` QueueManagerStatusCondition	Les conditions représentent les dernières observations disponibles de l'état du gestionnaire de files d'attente.
Tableau `endpoints` QueueManagerStatusEndpoint	Informations sur les noeuds finaux que ce gestionnaire de files d'attente expose, tels que les noeuds finaux de l'API ou de l'interface utilisateur.
Chaîne `name`	Nom du gestionnaire de files d'attente.
Chaîne `phase`	Phase de l'état du gestionnaire de files d'attente.
`versions` QueueManagerStatusVersion	Version de MQ utilisée et autres versions disponibles depuis le registre autorisé IBM.

.status.availability

Statut de disponibilité du gestionnaire de files d'attente.

Apparaît dans :

- «.status», à la page 148

Zone	Description
Valeur booléenne `initialQuorumEstablished`	Indique si un quorum initial a été établi pour NativeHA.

.status.conditions

QueueManagerStatusCondition définit les conditions du gestionnaire de files d'attente.

Apparaît dans :

- «.status», à la page 148

Zone	Description
Chaîne `lastTransitionTime`	Dernière transition de la condition d'un statut à un autre.
Chaîne `message`	Message lisible par l'utilisateur qui présente des détails sur la dernière transition.
Chaîne `reason`	Motif de la dernière transition de ce statut.
Chaîne `status`	Statut de la condition.
Chaîne `type`	Type de condition.

.status.endpoints

QueueManagerStatusEndpoint définit les noeuds finaux pour le gestionnaire de files d'attente.

Apparaît dans :

- «.status», à la page 148

Zone	Description
Chaîne `name`	Nom du noeud final.
Chaîne `type`	Type du nœud final, par exemple 'UI' pour un nœud final d'interface utilisateur, 'API' pour un nœud final d'API, 'OpenAPI' pour la documentation de l'API.
Chaîne `uri`	URI du noeud final.

.status.versions

Version de MQ utilisée et autres versions disponibles depuis le registre autorisé IBM.

Apparaît dans :

- «.status», à la page 148

Zone	Description
`available` QueueManagerStatusVersionAvailable	Autres versions de MQ disponibles depuis le registre autorisé IBM.

Zone	Description
Chaîne `reconciled`	Version spécifique d'IBM MQ qui est utilisée. Si une image personnalisée est spécifiée, cette valeur peut ne pas correspondre à la version de MQ réellement utilisée.

.status.versions.available

Autres versions de MQ disponibles depuis le registre autorisé IBM.

Apparaît dans :

- «.status.versions», à la page 149

Zone	Description
Tableau `channels`	Canaux disponibles pour la mise à jour automatique de la version de MQ.
Tableau `versions` Versions	Versions spécifiques de MQ qui sont disponibles.

.status.versions.available.versions

QueueManagerStatusVersion définit une version de MQ.

Apparaît dans :

- «.status.versions.available», à la page 150

Zone	Description
Chaîne `name`	Nom de cette version du gestionnaire de files d'attente. Valeurs valides pour la zone `spec.version`.

OpenShift CP4I *Conditions de statut de QueueManager (mq.ibm.com/v1beta1)*

Les zones **`status.conditions`** sont mises à jour pour refléter la condition de la ressource `QueueManager`. En général, les conditions décrivent des situations anormales. Un gestionnaire de files d'attente dans un état sain et prêt n'a pas de conditions **`Error`** ou **`Pending`** . Il peut comporter des conditions **`Warning`** de recommandation.

La prise en charge des conditions a été introduite dans IBM MQ Operator 1.2.

Les conditions suivantes sont définies pour une ressource `QueueManager` :

Tableau 1. Conditions de statut du gestionnaire de files d'attente

Composant	Type de condition	Code raison	Avertissement message
QueueManager[7]	En attente	Création	Le gestionnaire de files d'attente MQ est en cours de déploiement
	En attente	OidcPending	Le gestionnaire de files d'attente MQ attend l'enregistrement du client OIDC
	Erreur	Echec	Echec du déploiement du gestionnaire de files d'attente MQ
	Avertissement	UnsupportedVersion	[8]Un facteur a été installé par un opérateur qui n'est pas pris en charge sur la version OCP *<ocp_version>*. Ce facteur n'est pas pris en charge.
	Avertissement	EUSSupport	[9]Un facteur EUS *<mq_version>* a été installé mais est géré par un opérateur qui ne peut pas bénéficier de la durée de prise en charge étendue. Cet opérande ne correspond pas à la durée de prise en charge étendue.
	Avertissement	EUSSupport	[10]Un facteur EUS *<mq_version>* a été installé, mais la version OCP 4*<ocp_version>* ne se qualifie pas pour la durée de prise en charge étendue. Cet opérande ne correspond pas à la durée de prise en charge étendue.
	Avertissement	EUSSupport	[11]Un facteur EUS *<mq_version>* a été installé, mais la version OCP *<ocp_version>* n'est pas éligible pour la durée de prise en charge étendue. Cet opérande est pris en charge par une édition de CD standard.

[7] Les conditions `Creating` et `Failed` surveillent la progression globale du déploiement du gestionnaire de files d'attente. Si vous utilisez une licence IBM Cloud Pak for Integration et que la console Web MQ est activée, la condition `OidcPending` consigne le statut du gestionnaire de files d'attente lors de l'attente de l'enregistrement du client OIDC avec IAM.

Tableau 1. Conditions de statut du gestionnaire de files d'attente (suite)			
Composant	**Type de condition**	**Code raison**	**Avertissement message**
Pod[12]	En attente	PodPending	Pod pour le gestionnaire de files d'attente MQ en cours de déploiement
	Erreur	PodFailed	Pod pour le gestionnaire de files d'attente MQ en cours de déploiement
Mémoire[13]	En attente	StoragePending	Le stockage du gestionnaire de files d'attente MQ est mis à disposition
	Avertissement	StorageEphemeral	Utilisation du stockage temporaire pour un gestionnaire de files d'attente MQ de production
	Erreur	StorageFailed	Echec de la mise à disposition du stockage du gestionnaire de files d'attente MQ

Multi Génération de votre propre conteneur IBM MQ et code de déploiement

Développez un conteneur que vous avez généré vous-même. Il s'agit de la solution de conteneur la plus souple, qui exige toutefois de solides compétences relatives à la configuration des conteneurs et qui requiert que vous "possédiez" le conteneur résultant.

Avant de commencer

Avant de développer votre propre conteneur, déterminez si vous pouvez utiliser l'un des conteneurs préconditionnés fournis par IBM. Voir IBM MQ dans des conteneurs.

Pourquoi et quand exécuter cette tâche

Lorsque vous conditionnez IBM MQ en tant qu'image de conteneur, les modifications apportées à votre application peuvent être déployées sur des systèmes de test et de transfert rapidement et facilement, ce qui peut constituer un avantage majeur pour la distribution continue dans votre entreprise.

---

8 Opérateur 1.4.0 et versions ultérieures
9 Opérateur 1.4.0 et versions ultérieures
10 Opérateur 1.4.0 et versions ultérieures
11 Opérateur 1.3.0 uniquement
12 Les conditions du pod surveillent le statut des pods pendant le déploiement d'un gestionnaire de files d'attente. Si vous voyez une condition `PodFailed`, la condition globale du gestionnaire de files d'attente sera également définie sur `Failed`.
13 Les conditions de stockage surveillent la progression (condition `StoragePending`) des demandes de création de volumes pour le stockage permanent, et signalent les erreurs de liaison et autres échecs. Si une erreur se produit lors de l'allocation d'espace de stockage, la condition `StorageFailed` est ajoutée à la liste des conditions et la condition globale du gestionnaire de files d'attente sera également définie sur `Failed`.

Procédure

- «Planification de votre propre image de gestionnaire de files d'attente IBM MQ à l'aide d'un conteneur», à la page 153
- «Génération d'un exemple d'image de conteneur de gestionnaire de files d'attente IBM MQ», à la page 153
- «Exécution d'applications de liaison locale dans des conteneurs distincts», à la page 156

Concepts associés

IBM MQ dans des conteneurs

Multi Planification de votre propre image de gestionnaire de files d'attente IBM MQ à l'aide d'un conteneur

Vous devez tenir compte de plusieurs exigences lorsque vous exécutez un gestionnaire de files d'attente IBM MQ dans un conteneur. L'exemple d'image de conteneur répond à ces exigences, mais si vous voulez utiliser votre propre image, vous devez examiner la façon dont ces exigences sont traitées.

Supervision du processus

Lorsque vous exécutez un conteneur, vous exécutez principalement un processus unique (PID 1 dans le conteneur), qui peut ensuite engendrer des processus enfant.

Si le processus principal s'arrête, l'exécution du conteneur arrête le conteneur. Un gestionnaire de files d'attente IBM MQ requiert l'exécution de plusieurs processus en arrière-plan.

Par conséquent, vous devez vous assurer que votre processus principal reste actif tant que le gestionnaire de files d'attente est en cours d'exécution. Il est recommandé de vérifier que le gestionnaire de files d'attente est actif depuis ce processus, par exemple en émettant des requêtes administratives.

Remplissage de `/var/mqm`

Les conteneurs doivent être configurés avec `/var/mqm` en tant que volume.

Dans ce cas, le répertoire du volume est vide lorsque le conteneur démarre pour la première fois. En général, ce répertoire est rempli à l'installation, mais l'installation et l'exécution sont des environnements distincts dans le cadre de l'utilisation d'un conteneur.

Pour résoudre ce problème, lorsque votre conteneur démarre, vous pouvez utiliser la commande **`crtmqdir`** pour remplir `/var/mqm` lorsqu'il s'exécute pour la première fois.

Sécurité de conteneur

Pour réduire les exigences de sécurité de l'environnement d'exécution, les exemples d'image de conteneur sont installés à l'aide de l'installation décompressable d'IBM MQ. Ainsi, aucun bit `setuid` n'est défini et le conteneur n'a pas besoin d'utiliser l'escalade de privilèges. Certains systèmes de conteneur définissent les ID utilisateur que vous pouvez utiliser et l'installation décompressable ne fait aucune supposition concernant les utilisateurs du système d'exploitation disponibles.

Multi Génération d'un exemple d'image de conteneur de gestionnaire de files d'attente IBM MQ

Utilisez ces informations pour générer un exemple d'image de conteneur afin d'exécuter un gestionnaire de files d'attente IBM MQ dans un conteneur.

Pourquoi et quand exécuter cette tâche

Tout d'abord, vous générez une image de base contenant un système de fichiers Red Hat Universal Base Image et une installation propre d'IBM MQ.

Ensuite, vous générez une autre couche d'image de conteneur sur la base, qui ajoute une configuration IBM MQ assurant une sécurité de base par ID utilisateur et mot de passe.

Enfin, vous exécutez un conteneur à l'aide de cette image comme système de fichiers, avec le contenu du répertoire `/var/mqm` fourni par un volume spécifique au conteneur sur le système de fichiers hôte.

Procédure

- Pour des informations sur la génération d'un exemple d'image de conteneur pour l'exécution d'un gestionnaire de files d'attente IBM MQ dans un conteneur, voir les sous-rubriques suivantes :
  - «Génération d'un exemple d'image de gestionnaire de files d'attente IBM MQ de base», à la page 154
  - «Génération d'un exemple d'image de gestionnaire de files d'attente IBM MQ configurée», à la page 154

Multi Génération d'un exemple d'image de gestionnaire de files d'attente IBM MQ de base

Pour utiliser IBM MQ dans votre propre image de conteneur, vous devez d'abord générer une image de base avec une installation propre d'IBM MQ. Les étapes ci-dessous expliquent comment générer un exemple d'image de base à l'aide d'un exemple de code hébergé sur GitHub.

Procédure

- Utilisez les fichiers make fournis dans le référentiel GitHub du conteneur mq pour générer l'image de conteneur de production.

  Suivez les instructions de la section Génération d'une image de conteneur sur GitHub. Si vous prévoyez de configurer un accès sécurisé à l'aide de la contrainte de contexte de sécurité (SCC) Red Hat OpenShift Container Platform "restreinte" , vous devez utiliser le package IBM MQ 'No-Install'.

Résultats

A présent, vous disposez d'une image de conteneur de base dans laquelle IBM MQ est installé.

Vous êtes maintenant prêt à générer un exemple d'image de gestionnaire de files d'attente IBM MQ configurée.

Multi Génération d'un exemple d'image de gestionnaire de files d'attente IBM MQ configurée

Une fois que vous avez généré l'image de conteneur IBM MQ de base générique, vous devez appliquer votre propre configuration pour autoriser l'accès sécurisé. Pour ce faire, créez votre propre couche d'image de conteneur en utilisant l'image générique comme parent.

Avant de commencer

V 9.2.0 Cette tâche suppose que, lorsque vous avez créé votre exemple d'image de gestionnaire de files d'attente IBM MQ de base, vous avez utilisé le package "No-Install" IBM MQ. Sinon, vous ne pouvez pas configurer l'accès sécurisé à l'aide de la contrainte de contexte de sécurité (SCC) Red Hat OpenShift Container Platform "restreinte" . La contrainte SCC "restricted", qui est utilisée par défaut, utilise des ID utilisateur aléatoires et empêche l'escalade des privilèges en passant à un autre utilisateur. Le programme d'installation traditionnel basé sur RPM IBM MQ repose sur un utilisateur et un groupe `mqm` et utilise également des bits `setuid` sur des programmes exécutables. Dans IBM MQ 9.2, lorsque vous utilisez le package "No-Install" IBM MQ, il n'y a plus d'utilisateur `mqm`, ni de groupe `mqm`.

Procédure

1. Créez un répertoire et ajoutez un fichier nommé `config.mqsc` dont le contenu est le suivant :

```
DEFINE QLOCAL(EXAMPLE.QUEUE.1) REPLACE
```

   Notez que l'exemple précédent utilise une authentification simple par ID utilisateur et mot de passe. Toutefois, vous pouvez appliquer toute configuration de sécurité requise par votre entreprise.

2. Créez un fichier nommé `Dockerfile` dont le contenu est le suivant :

```
FROM mq
COPY config.mqsc /etc/mqm/
```

3. Générez votre image de conteneur personnalisée avec la commande suivante :

```
docker build -t mymq .
```

   Où "`.`" est le répertoire contenant les deux fichiers que vous venez de créer.

   Docker créé ensuite un conteneur temporaire à l'aide de cette image, et exécute les commandes restantes.

   Remarque : Sous Red Hat Enterprise Linux (RHEL), vous utilisez la commande **`docker`** (RHEL V7) ou **podman** (RHEL V7 ou RHEL V8). Sous Linux, vous devez exécuter des commandes **`docker`** en indiquant **sudo** au début de la commande afin d'obtenir des privilèges supplémentaires.

4. Exécutez votre nouvelle image personnalisée afin de créer un nouveau conteneur avec l'image de disque que vous venez de créer.

   Votre nouvelle couche d'image ne spécifie pas de commande particulière à exécuter ; par conséquent, la commande est héritée de l'image parent. Le point d'entrée du parent (code disponible sur GitHub) :

   - Crée un gestionnaire de files d'attente
   - Démarre le gestionnaire de files d'attente
   - Crée un programme d'écoute par défaut
   - Exécutez ensuite les commandes MQSC à partir de `/etc/mqm/config.mqsc`.

   Emettez les commandes suivantes pour exécuter votre nouvelle image personnalisée :

```
docker run \
  --env LICENSE=accept \
  --env MQ_QMGR_NAME=QM1 \
  --volume /var/example:/var/mqm \
  --publish 1414:1414 \
  --detach \
  mymq
```

   où :

   Le premier paramètre env
   Transmet une variable d'environnement dans le conteneur, qui reconnaît votre acceptation de la licence pour IBM IBM WebSphere MQ. Vous pouvez aussi définir la variable LICENSE afin d'afficher la licence.

   Voir Informations sur les licences IBM MQ pour plus de détails sur les licences d'IBM MQ.

   Le deuxième paramètre env
   Définit le nom du gestionnaire de files d'attente que vous utilisez.

   Le paramètre volume
   Indique que le conteneur que MQ écrit dans `/var/mqm` doit en fait être écrit sur `/var/example` sur l'hôte.

   Cette option signifie qu'il est facile de supprimer le conteneur ultérieurement tout en conservant les données persistantes. Elle facilite également l'affichage des fichiers journaux.

Le paramètre publish
Mappe des ports du système hôte à des ports dans le conteneur. Le conteneur s'exécute par défaut avec sa propre adresse IP interne, ce qui signifie que vous devez mapper spécifiquement tout port que vous voulez exposer.

Dans cet exemple, cela signifie que vous devez mapper le port 1414 sur l'hôte au port 1414 dans le conteneur.

Le paramètre detach
Exécute le conteneur en arrière-plan.

Résultats

Vous avez généré une image de conteneur configurée et pouvez afficher les conteneurs en cours d'exécution avec la commande docker **`docker ps`**. Vous pouvez afficher les processus IBM MQ qui s'exécutent dans votre conteneur avec la commande **`docker top`**.

Avertissement :

Vous pouvez afficher les journaux d'un conteneur avec la commande **`docker logs ${CONTAINER_ID}`**.

Que faire ensuite

- Si votre conteneur ne s'affiche pas lorsque vous utilisez la commande **`docker ps`**, il se peut que le conteneur soit défaillant. Vous pouvez voir les conteneurs ayant échoué à l'aide de la commande **`docker ps -a`** .
- Lorsque vous utilisez la commande **`docker ps -a`**, l'ID de conteneur est affiché. Il l'est également lorsque vous émettez la commande **`docker run`**.
- Vous pouvez afficher les journaux d'un conteneur avec la commande **`docker logs ${CONTAINER_ID}`**.

Multi Exécution d'applications de liaison locale dans des conteneurs distincts

Avec le partage d'espace de nom de processus entre des conteneurs dans Docker, vous pouvez exécuter des applications qui requièrent une connexion de liaison locale à IBM MQ dans des conteneurs distincts depuis le gestionnaire de files d'attente IBM MQ.

Pourquoi et quand exécuter cette tâche

Cette fonctionnalité est prise en charge dans IBM MQ 9.0.3 et dans les gestionnaires de files d'attente ultérieurs.

Vous devez respecter les restrictions suivantes :

- Vous devez partager l'espace de nom PID des conteneurs avec l'argument `--pid`.
- Vous devez partager l'espace de nom IPC des conteneurs avec l'argument `--ipc`.
- Vous devez :
  1. Partager l'espace de nom UTS des conteneurs avec l'hôte avec l'argument `--uts` ou
  2. Vous assurer que les conteneurs possèdent le même nom d'hôte avec l'argument `-h` ou `--hostname`.
- Vous devez monter le répertoire de données IBM MQ dans un volume disponible pour tous les conteneurs sous le répertoire `/var/mqm`.

Vous pouvez essayer cette fonctionnalité en effectuant les étapes ci-après sur un système Linux sur lequel Docker est déjà installé.

L'exemple ci-dessous utilise l'exemple d'image de conteneur IBM MQ. Vous trouverez les détails de cette image sur Github.

Procédure

1. Créez un répertoire temporaire qui servira de volume en émettant la commande suivante :

```
mkdir /tmp/dockerVolume
```

2. Créez un gestionnaire de files d'attente (QM1) dans un conteneur, avec le nom `sharedNamespace`, en émettant la commande suivante :

```
docker run -d -e LICENSE=accept -e MQ_QMGR_NAME=QM1 --volume /tmp/dockerVol:/mnt/mqm
--uts host --name sharedNamespace ibmcom/mq
```

3. Démarrez un deuxième conteneur nommé `secondaryContainer`, qui repose sur `ibmcom/mq`, sans créer de gestionnaire de files d'attente, en émettant la commande suivante :

```
docker run --entrypoint /bin/bash --volumes-from sharedNamespace --pid
container:sharedNamespace --ipc container:sharedNamespace --uts host --name
secondaryContainer -it --detach ibmcom/mq
```

4. Exécutez la commande **dspmq** dans le deuxième conteneur pour afficher le statut des deux gestionnaires de files d'attente en émettant la commande suivante :

```
docker exec secondaryContainer dspmq
```

5. Exécutez la commande suivante afin de traiter les commandes MQSC pour le gestionnaire de files d'attente s'exécutant dans l'autre conteneur :

```
docker exec -it secondaryContainer runmqsc QM1
```

Résultats

Désormais, vous disposez d'applications locales qui s'exécutent dans des conteneurs distincts et vous pouvez exécuter des commandes telles que **dspmq**, **amqsput**, **amqsget** et **runmqsc** en tant que liaisons locales pour le gestionnaire de files d'attente QM1 depuis le deuxième conteneur.

Si les résultats ne sont pas ceux que vous attendiez, voir «Traitement des incidents liés à vos applications d'espace de nom», à la page 157 pour plus d'informations.

Multi Traitement des incidents liés à vos applications d'espace de nom

Lorsque vous utilisez des espaces de nom partagés, vous devez vous assurer que vous partagez tous les espaces de nom (IPC, PID et UTS/nom d'hôte) et tous les volumes montés ; si tel n'est pas le cas, vos applications ne fonctionneront pas.

Voir «Exécution d'applications de liaison locale dans des conteneurs distincts», à la page 156 pour la liste des restrictions à respecter.

Si votre application ne répond pas à toutes les restrictions répertoriées, il se peut que vous rencontriez des problèmes. Par exemple, le conteneur pourra démarrer, mais la fonctionnalité que vous attendez ne fonctionnera pas.

La liste ci-après met en évidence certaines causes communes et le comportement qui peut découler du non-respect de l'une des restrictions.

- Si vous oubliez de partager l'espace de nom (UTS/PID/IPC) ou le nom d'hôte des conteneurs et que vous montez le volume, votre conteneur pourra voir le gestionnaire de files d'attente mais ne pourra pas interagir avec lui.

– Pour les commandes **dspmq**, le code suivant s'affiche :

```
docker exec container dspmq

QMNAME(QM1)                       STATUS(Status not available)
```

– Pour les commandes **runmqsc** ou d'autres commandes qui tentent d'établir la connexion au gestionnaire de files d'attente, vous êtes susceptible de recevoir le message d'erreur AMQ8146 :

```
docker exec -it container runmqsc QM1
5724-H72 (C) Copyright IBM Corp. 1994, 2024.
Starting MQSC for queue manager QM1.
AMQ8146: IBM MQ queue manager not available
```

- Si vous partagez tous les espaces de nom requis mais que vous ne montez pas de volume partagé dans le répertoire `/var/mqm` et que vous disposez d'un chemin de données IBM MQ valide, vos commandes reçoivent également des messages d'erreur AMQ8146.

  Toutefois, **dspmq** ne peut pas voir votre gestionnaire de files d'attente et il renvoie une réponse vierge à la place :

```
docker exec container dspmq
```

- Si vous partagez tous les espaces de nom requis, mais que vous ne montez pas de volume partagé dans le répertoire `/var/mqm` et que vous ne disposez pas d'un chemin de données IBM MQ valide (ou d'un chemin de données IBM MQ), plusieurs erreurs se produisent, car le chemin de données est un composant clé d'une installation IBM MQ. Sans le chemin d'accès aux données, IBM MQ ne peut pas fonctionner.

  Si vous exécutez l'une des commandes suivantes et que des réponses similaires aux exemples sont affichées, vérifiez que vous avez monté le répertoire ou créé un répertoire de données IBM MQ :

```
docker exec container dspmq
'No such file or directory' from /var/mqm/mqs.ini
AMQ6090: IBM MQ was unable to display an error message FFFFFFFF.
AMQffff

docker exec container dspmqver
AMQ7047: An unexpected error was encountered by a command. Reason code is 0.

docker exec container mqrc
<file path>/mqrc.c[1152]
lpiObtainQMDetails --> 545261715

docker exec container crtmqm QM1
AMQ8101: IBM MQ error (893) has occurred.

docker exec container strmqm QM1
AMQ6239: Permission denied attempting to access filesystem location '/var/mqm'.
AMQ7002: An error occurred manipulating a file.

docker exec container endmqm QM1
AMQ8101: IBM MQ error (893) has occurred.

docker exec container dltmqm QM1
AMQ7002: An error occurred manipulating a file.

docker exec container strmqweb
<file path>/mqrc.c[1152]
lpiObtainQMDetails --> 545261715
```

CP4I Création du groupe Native HA si vous créez vos propres conteneurs

Vous devez créer, configurer et démarrer trois gestionnaires de files d'attente pour créer le groupe Native HA.

Pourquoi et quand exécuter cette tâche

La méthode recommandée pour créer une solution Native HA consiste à utiliser l'opérateur IBM MQ (voir Native HA). Sinon, si vous créez vos propres conteneurs, vous pouvez suivre ces instructions.

Pour créer un groupe Native HA, vous créez trois gestionnaires de files d'attente sur trois noeuds dont le type de journal est défini sur `log replication`. Vous éditez ensuite le fichier `qm.ini` pour chaque gestionnaire de files d'attente afin d'ajouter les détails de connexion pour chacun des trois noeuds afin qu'ils puissent répliquer les données de journal les uns sur les autres.

Vous devez ensuite démarrer les trois gestionnaires de files d'attente afin qu'ils puissent vérifier que les trois instances peuvent communiquer entre elles et déterminer laquelle d'entre elles sera l'instance active et laquelle sera les répliques.

Procédure

1. Sur chacun des trois noeuds, créez un gestionnaire de files d'attente, en spécifiant un type de journal de réplique de journal et en fournissant un nom unique pour chaque instance de journal. Chaque gestionnaire de files d'attente porte le même nom:

   ```
   crtmqm -lr instance_name qmname
   ```

   Exemple :

   ```
   node 1> crtmqm -lr qm1_inst1 qm1

   node 2> crtmqm -lr qm1_inst2 qm1

   node 3> crtmqm -lr qm1_inst3 qm1
   ```

2. Lorsque la création de chaque gestionnaire de files d'attente aboutit, une section supplémentaire nommée `NativeHALocalInstance` est ajoutée au fichier de configuration du gestionnaire de files d'attente, `qm.ini`. Un attribut `Name` est ajouté à la section spécifiant le nom d'instance fourni.

   Vous pouvez éventuellement ajouter les attributs suivants à la strophe `NativeHALocalInstance` dans le fichier `qm.ini` :

   `KeyRepository`
   Emplacement du référentiel de clés qui contient le certificat numérique à utiliser pour la protection du trafic de réplication des journaux. L'emplacement est donné au format de radical, c'est-à-dire qu'il inclut le chemin d'accès complet et le nom de fichier sans extension. Si l'attribut de section `KeyRepository` est omis, les données de réplication de journal sont échangées entre les instances en texte en clair.

   `CertificateLabel`
   Libellé de certificat identifiant le certificat numérique à utiliser pour la protection du trafic de réplication des journaux. Si `KeyRepository` est fourni mais que `CertificateLabel` est omis, la valeur par défaut `ibmwebspheremq`*`queue_manager`* est utilisée.

   `CipherSpec`
   Le MQ CipherSpec à utiliser pour protéger le trafic de réplication des journaux. Si cet attribut de section est fourni, `KeyRepository` doit également être fourni. Si `KeyRepository` est fourni mais que `CipherSpec` est omis, la valeur par défaut ANY est utilisée.

   `LocalAddress`
   Adresse de l'interface réseau locale qui accepte le trafic de réplication de journal. Si cet attribut de section est fourni, il identifie l'interface réseau locale et / ou le port en utilisant le format " [addr] [(port)]. L'adresse réseau peut être spécifiée sous la forme d'un nom d'hôte, IPv4 à notation décimale à point ou IPv6 au format hexadécimal. Si cet attribut est omis, le gestionnaire de files d'attente tente de se connecter à toutes les interfaces réseau et utilise le port spécifié dans `ReplicationAddress` dans la section `NativeHAInstances` correspondant au nom de l'instance locale.

HeartbeatInterval
L'intervalle des pulsations définit la fréquence en millisecondes à laquelle une instance active d'un gestionnaire de files d'attente Native HA envoie une pulsation réseau. Il est compris entre 500 (0,5 secondes) et 60000 (1 minute). Une valeur hors de cette plage empêche le démarrage du gestionnaire de files d'attente. Si cet attribut est omis, une valeur par défaut de 5000 (5 secondes) est utilisée. Chaque instance doit utiliser le même intervalle de pulsations.

HeartbeatTimeout
Le dépassement du délai d'attente du signal de présence définit le temps pendant lequel une réplique d'instance d'un gestionnaire de files d'attente Native HA attend avant de considérer que l'instance active ne répondra pas. Cette valeur doit être comprise entre 500 (0,5 secondes) et 120000 (2 minutes). La valeur du dépassement du délai d'attente du signal de présence doit être supérieure ou égale à celle de l'intervalle des pulsations.

Une valeur non valide empêche le démarrage du gestionnaire de files d'attente. Si cet attribut est omis, une réplique attend 2 x `HeartbeatInterval` avant de lancer le processus pour sélectionner une nouvelle instance active. Chaque instance doit utiliser la même valeur de dépassement du délai d'attente du signal de présence.

RetryInterval
L'intervalle entre les nouvelles tentatives définit la fréquence en millisecondes à laquelle un gestionnaire de files d'attente Native HA doit retenter un lien de réplication défectueux. Cet intervalle doit être compris entre 500 (0,5 secondes) et 120000 (2 minutes). Si cet attribut est omis, une réplique attend 2 x `HeartbeatInterval` avant de réessayer un lien de réplication ayant échoué.

3. Editez le fichier `qm.ini` pour chaque gestionnaire de files d'attente et ajoutez les détails de connexion. Vous ajoutez trois sections `NativeHAInstance` , une pour chaque instance de gestionnaire de files d'attente dans le groupe Native HA (y compris l'instance locale). Ajoutez les attributs suivants:

   Nom
   Indiquez le nom d'instance que vous avez utilisé lors de la création de l'instance de gestionnaire de files d'attente.

   ReplicationAddress
   Indiquez le nom d'hôte, IPv4 décimale à point ou IPv6 adresse au format hexadécimal de l'instance. Vous pouvez spécifier l'adresse en tant que nom d'hôte, IPv4 en notation décimale à point ou IPv6 en format hexadécimal. L'adresse de réplication doit pouvoir être résolue et routable à partir de chaque instance du groupe. Le numéro de port à utiliser pour la réplication de journal doit être indiqué entre crochets, par exemple:

   ```
   ReplicationAddress=host1.example.com(4444)
   ```

   Remarque : Les sections `NativeHAInstance` sont identiques sur chaque instance et peuvent être fournies à l'aide de la configuration automatique (**`crtmqm -ii`**).

4. Démarrez chacune des trois instances:

   ```
   strmqm QMgrName
   ```

   Lorsque les instances sont démarrées, elles communiquent pour vérifier que les trois instances sont en cours d'exécution, puis décident laquelle des trois instances est l'instance active, tandis que les deux autres instances continuent de s'exécuter en tant que répliques.

Exemple

L'exemple suivant illustre la section d'un fichier `qm.ini` spécifiant les détails Native HA requis pour l'une des trois instances:

```
NativeHALocalInstance:
  LocalName=node-1

NativeHAInstance:
  Name=node-1
```

```
  ReplicationAddress=host1.example.com(4444)
NativeHAInstance:
  Name=node-2
  ReplicationAddress=host2.example.com(4444)
NativeHAInstance:
  Name=node-3
  ReplicationAddress=host3.example.com(4444)
```

Kubernetes Remarques sur l'exécution de votre propre mise à jour en continu d'un gestionnaire de files d'attente natif de haute disponibilité

Toute mise à jour de la version IBM MQ ou de la spécification Pod pour un gestionnaire de files d'attente natif de haute disponibilité, vous demandera d'effectuer une mise à jour en continu des instances du gestionnaire de files d'attente. IBM MQ Operator gère cela automatiquement, mais si vous construisez votre propre code de déploiement, il y a des considérations importantes à prendre en compte.

Remarque : Le fichier Exemple de graphique Helm inclut un script de shell pour effectuer une mise à jour en continu, mais le script n'est **pas** adapté à l'utilisation de la production, car il n'aborde pas les considérations de cette rubrique.

Dans Kubernetes, les ressources `StatefulSet` sont utilisées pour gérer les mises à jour de démarrage et en continu commandées. Une partie de la procédure de démarrage consiste à démarrer chaque Pod individuellement, à attendre qu'il devienne prêt, puis à passer à la prochaine Pod. Cela ne fonctionnera pas pour Native HA, car tous les pods doivent être démarrés pour qu'ils puissent effectuer une élection de leader. Par conséquent, la zone `.spec.podManagementPolicy` sur le `StatefulSet` doit être définie sur `Parallel`. Cela signifie également que tous les Pods seront également mis à jour en parallèle, ce qui est particulièrement indésirable. Pour cette raison, le `StatefulSet` doit également utiliser la stratégie de mise à jour `OnDelete`.

L'inaptitude à utiliser le code de mise à jour en continu `StatefulSet` entraîne un besoin de code de mise à jour en continu personnalisé, qui doit prendre en compte les éléments suivants :

- Procédure générale de mise à jour en continu
- Réduire le temps d'indisponibilité en mettant à jour les Pods dans le meilleur ordre
- Traitement des modifications dans l'état du cluster
- Traitement des erreurs
- Traitement des problèmes de temps

Procédure générale de mise à jour en continu

Le code de mise à jour en continu doit attendre que chaque instance affiche un statut de `REPLICA` à partir de `dspmq`. Cela signifie que l'instance a exécuté un certain niveau de démarrage (par exemple, le conteneur est démarré et les processus MQ sont en cours d'exécution), mais qu'elle n'a pas encore réussi à parler aux autres instances. Par exemple, le pod A est redémarré et dès qu'il est à l'état `REPLICA` , le pod B est redémarré. Une fois que Pod B commence par la nouvelle configuration, il devrait être capable de parler à Pod A, et peut former le quorum, et soit A ou B deviendra la nouvelle instance active.

Dans ce cas, il est utile d'avoir un délai après que chaque Pod a atteint l'état `REPLICA`, afin de lui permettre de se connecter à ses homologues et d'établir le quorum.

Réduire le temps d'indisponibilité en mettant à jour les Pods dans le meilleur ordre

Le code de mise à jour en continu doit supprimer les Pods un à la fois, en commençant par les Pods qui se trouvent dans un état d'erreur connu, suivis des Pods qui n'ont pas démarré avec succès. Le gestionnaire Pod de files d'attente actif doit généralement être mis à jour en dernier.

Il est également important de mettre en pause la suppression des Pods si la dernière mise à jour a donné lieu à un Pod dans un état d'erreur connu. Cela empêche le déploiement d'une mise à jour interrompue sur tous les Pods. Par exemple, cela peut se produire si le Pod est mis à jour pour utiliser une nouvelle image de conteneur qui n'est pas accessible (ou contient une typo).

Traitement des modifications dans l'état du cluster

Le code de mise à jour en continu doit réagir de manière appropriée aux changements en temps réel dans l'état du cluster. Par exemple, l'un des Pods du gestionnaire de files d'attente peut être expulsé en raison d'un réamorçage du noeud ou de la pression du noeud. Il est possible qu'un Pod expulsé ne soit pas immédiatement reprogrammée si le cluster est occupé. Dans ce cas, le code de mise à jour en continu doit attendre correctement avant de redémarrer les autres Pods.

Traitement des erreurs

Le code de mise à jour en continu doit être robuste pour les échecs lors de l'appel de l'API de Kubernetes et d'autres comportements de cluster inattendus.

En outre, le code de mise à jour en continu lui-même doit être tolérant pour être redémarré. Une mise à jour en continu peut être longue et le code doit être redémarré.

Traitement des problèmes de temps

Le code de mise à jour en continu doit vérifier les révisions de mise à jour du Pod, de sorte qu'il puisse s'assurer que le Pod ait redémarré. Cela permet d'éviter les problèmes de temps où un Pod peut indiquer qu'il est « Démarré », mais n'est pas encore terminé en fait.

Concepts associés

«Comment utiliser IBM MQ dans des conteneurs», à la page 5
Il existe plusieurs options d'utilisation de IBM MQ dans des conteneurs : vous pouvez choisir d'utiliser le IBM MQ Operator, qui utilise des images de conteneur pré-conditionnées, ou créer vos propres images et code de déploiement.

CP4I Affichage du statut des gestionnaires de files d'attente Native HA pour les conteneurs personnalisés

Pour les conteneurs personnalisés, vous pouvez afficher le statut des instances Native HA à l'aide de la commande **dspmq** .

Pourquoi et quand exécuter cette tâche

Vous pouvez utiliser la commande **dspmq** pour afficher le statut opérationnel d'une instance de gestionnaire de files d'attente sur un noeud. Les informations renvoyées varient selon que l'instance est active ou qu'il s'agit d'une réplique. Les informations fournies par l'instance active sont définitives, tandis que celles des noeuds de réplique peuvent être obsolètes.

Vous pouvez effectuer les actions suivantes :

- Déterminer si l'instance de gestionnaire de files d'attente sur le noeud actuel est active ou s'il s'agit d'une réplique.
- Afficher le statut Native HA opérationnel de l'instance sur le noeud actuel.
- Afficher le statut opérationnel des trois instances dans une configuration Native HA.

Les zones de statut suivantes sont utilisées pour signaler le statut de la configuration Native HA :

ROLE
Indique le rôle en cours de l'instance et est l'un des rôles `Active`, `Replica` ou `Unknown`.

INSTANCE
Nom fourni pour cette instance du gestionnaire de files d'attente lorsque ce dernier a été créé à l'aide de l'option **-lr** de la commande **crtmqm**.

INSYNC
Indique si l'instance peut prendre la relève en tant qu'instance active, si nécessaire.

QUORUM
Indique le statut de quorum au format *nombre_instances_synchronisées*/*nombre_instances_configurées*.

REPLADDR
Adresse de réplication de l'instance de gestionnaire de files d'attente.

CONNACTV
Indique si le noeud est connecté à l'instance active.

BACKLOG
Indique le nombre de kilooctets de retard de l'instance.

CONNINST
Indique si l'instance désignée est connectée à cette instance.

ALTDATE
Indique la date à laquelle ces informations ont été mises à jour pour la dernière fois (vide si elles n'ont jamais été mises à jour).

ALTTIME
Indique l'heure à laquelle ces informations ont été mises à jour pour la dernière fois (vide si elles n'ont jamais été mises à jour).

Procédure

- Pour déterminer si une instance de gestionnaire de files d'attente est exécutée comme instance active ou comme réplique :

  ```
  dspmq -o status -m QMgrName
  ```

  Une instance active d'un gestionnaire de files d'attente nommé BOB signale le statut suivant :

  ```
  QMNAME(BOB)                 STATUS(Running)
  ```

  Une réplique d'instance d'un gestionnaire de files d'attente nommé BOB signale le statut suivant :

  ```
  QMNAME(BOB)                 STATUS(Replica)
  ```

  Une instance inactive signale le statut suivant :

  ```
  QMNAME(BOB)                 STATUS(Ended Immediately)
  ```

- Pour déterminer le statut opérationnel Native HA de l'instance sur le noeud en cours:

  ```
  dspmq -o nativeha -m QMgrName
  ```

  L'instance active d'un gestionnaire de files d'attente nommé BOB peut signaler le statut suivant :

  ```
  QMNAME(BOB)               ROLE(Active) INSTANCE(inst1) INSYNC(Yes) QUORUM(3/3)
  ```

  Une réplique d'instance d'un gestionnaire de files d'attente nommé BOB peut signaler le statut suivant :

  ```
  QMNAME(BOB)               ROLE(Replica) INSTANCE(inst2) INSYNC(Yes) QUORUM(2/3)
  ```

  Une instance inactive d'un gestionnaire de files d'attente nommé BOB peut signaler le statut suivant :

  ```
  QMNAME(BOB)               ROLE(Unknown) INSTANCE(inst3) INSYNC(no) QUORUM(0/3)
  ```

- Pour déterminer le statut Native HA opérationnel de toutes les instances de la configuration Native HA :

  ```
  dspmq -o nativeha -x -m QMgrName
  ```

Si vous exécutez cette commande sur le noeud qui exécute l'instance active du gestionnaire de files d'attente BOB, vous risquez de recevoir le statut suivant :

```
QMNAME(BOB)            ROLE(Active) INSTANCE(inst1) INSYNC(Yes) QUORUM(3/3)
 INSTANCE(inst1) ROLE(Active)  REPLADDR(9.20.123.45) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
 INSTANCE(inst2) ROLE(Replica) REPLADDR(9.20.123.46) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
 INSTANCE(inst3) ROLE(Replica) REPLADDR(9.20.123.47) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
```

Si vous exécutez cette commande sur un noeud qui exécute une réplique d'instance du gestionnaire de files d'attente BOB, vous risquez de recevoir le statut suivant, qui indique que l'une des répliques est en retard :

```
QMNAME(BOB)            ROLE(Replica) INSTANCE(inst2) INSYNC(Yes) QUORUM(2/3)
 INSTANCE(inst2) ROLE(Replica) REPLADDR(9.20.123.46) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
 INSTANCE(inst1) ROLE(Active)  REPLADDR(9.20.123.45) CONNACTV(Yes) INSYNC(Yes) BACKLOG(0)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
 INSTANCE(inst3) ROLE(Replica) REPLADDR(9.20.123.47) CONNACTV(Yes) INSYNC(No)  BACKLOG(435)
CONNINST(Yes) ALTDATE(2022-01-12) ALTTIME(12.03.44)
```

Si vous exécutez cette commande sur un noeud qui exécute une instance inactive du gestionnaire de files d'attente BOB, vous risquez de recevoir le statut suivant :

```
QMNAME(BOB)            ROLE(Unknown) INSTANCE(inst3) INSYNC(no) QUORUM(0/3)
 INSTANCE(inst1) ROLE(Unknown) REPLADDR(9.20.123.45) CONNACTV(Unknown) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ALTDATE() ALTTIME()
 INSTANCE(inst2) ROLE(Unknown) REPLADDR(9.20.123.46) CONNACTV(Unknown) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ALTDATE() ALTTIME()
 INSTANCE(inst3) ROLE(Unknown) REPLADDR(9.20.123.47) CONNACTV(No) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ALTDATE() ALTTIME()
```

Si vous exécutez la commande alors que les instances sont encore en cours de négociation pour déterminer l'instance active et les répliques, vous recevez le statut suivant :

```
QMNAME(BOB)              STATUS(Negotiating)
```

Référence associée

dspmq

CP4I Arrêt des gestionnaires de files d'attente Native HA

Vous pouvez utiliser la commande **endmqm** pour arrêter un gestionnaire de files d'attente actif ou de réplique faisant partie d'un groupe Native HA.

Procédure

- Pour arrêter l'instance active d'un gestionnaire de files d'attente, voir Arrêt des gestionnaires de files d'attente Native HA dans la section Configuration de cette documentation.

Remarques

:NONE.

Le présent document peut contenir des informations ou des références concernant certains produits, logiciels ou services IBM non annoncés dans ce pays. Pour plus de détails, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial IBM. Toute référence à un produit, logiciel ou service IBM n'implique pas que seul ce produit, logiciel ou service IBM puisse être utilisé. Tout autre élément fonctionnellement équivalent peut être utilisé, s'il n'enfreint aucun droit d'IBM. Il est de la responsabilité de l'utilisateur d'évaluer et de vérifier lui-même les installations et applications réalisées avec des produits, logiciels ou services non expressément référencés par IBM.

IBM peut détenir des brevets ou des demandes de brevet couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous donne aucun droit de licence sur ces brevets ou demandes de brevet. Si vous désirez recevoir des informations concernant l'acquisition de licences, veuillez en faire la demande par écrit à l'adresse suivante :

IBM EMEA Director of Licensing
IBM Corporation
Tour Descartes
Armonk, NY 10504-1785
U.S.A.

Pour toute demande d'informations relatives au jeu de caractères codé sur deux octets, contactez le service de propriété intellectuelle IBM ou envoyez vos questions par courrier à l'adresse suivante :

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japon

Le paragraphe suivant ne s’applique ni au Royaume-Uni, ni dans aucun pays dans lequel il serait contraire aux lois locales. LE PRESENT DOCUMENT EST LIVRE "EN L'ETAT" SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES INFORMATIONS EN CAS DE CONTREFACON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE. Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

Le présent document peut contenir des inexactitudes ou des coquilles. Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. IBM peut, à tout moment et sans préavis, modifier les produits et logiciels décrits dans ce document.

Les références à des sites Web non IBM sont fournies à titre d'information uniquement et n'impliquent en aucun cas une adhésion aux données qu'ils contiennent. Les éléments figurant sur ces sites Web ne font pas partie des éléments du présent produit IBM et l'utilisation de ces sites relève de votre seule responsabilité.

IBM pourra utiliser ou diffuser, de toute manière qu'elle jugera appropriée et sans aucune obligation de sa part, tout ou partie des informations qui lui seront fournies.

Les licenciés souhaitant obtenir des informations permettant : (i) l'échange des données entre des logiciels créés de façon indépendante et d'autres logiciels (dont celui-ci), et (ii) l'utilisation mutuelle des données ainsi échangées, doivent adresser leur demande à :

IBM Corporation
Coordinateur d'interopérabilité logicielle, département 49XA
3605 Autoroute 52 N

Rochester, MN 55901
U.S.A.

Ces informations peuvent être soumises à des conditions particulières, prévoyant notamment le paiement d'une redevance.

Le logiciel sous licence décrit dans le présent document et tous les éléments sous disponibles s'y rapportant sont fournis par IBM conformément aux dispositions du Contrat sur les produits et services IBM, aux Conditions Internationales d'Utilisation de Logiciels IBM ou de tout autre accord équivalent.

Les données de performance indiquées dans ce document ont été déterminées dans un environnement contrôlé. Par conséquent, les résultats peuvent varier de manière significative selon l'environnement d'exploitation utilisé. Certaines mesures évaluées sur des systèmes en cours de développement ne sont pas garanties sur tous les systèmes disponibles. En outre, elles peuvent résulter d'extrapolations. Les résultats peuvent donc varier. Il incombe aux utilisateurs de ce document de vérifier si ces données sont applicables à leur environnement d'exploitation.

Les informations concernant des produits non IBM ont été obtenues auprès des fournisseurs de ces produits, par l'intermédiaire d'annonces publiques ou via d'autres sources disponibles. IBM n'a pas testé ces produits et ne peut confirmer l'exactitude de leurs performances ni leur compatibilité. Elle ne peut recevoir aucune réclamation concernant des produits non IBM. Toute question concernant les performances de produits non IBM doit être adressée aux fournisseurs de ces produits.

Toute instruction relative aux intentions d'IBM pour ses opérations à venir est susceptible d'être modifiée ou annulée sans préavis, et doit être considérée uniquement comme un objectif.

Le présent document peut contenir des exemples de données et de rapports utilisés couramment dans l'environnement professionnel. Ces exemples mentionnent des noms fictifs de personnes, de sociétés, de marques ou de produits à des fins illustratives ou explicatives uniquement. Toute ressemblance avec des noms de personnes, de sociétés ou des données réelles serait purement fortuite.

Licence sur les droits d'auteur :

Le présent logiciel contient des exemples de programmes d'application en langage source destinés à illustrer les techniques de programmation sur différentes plateformes d'exploitation. Vous avez le droit de copier, de modifier et de distribuer ces exemples de programmes sous quelque forme que ce soit et sans paiement d'aucune redevance à IBM, à des fins de développement, d'utilisation, de vente ou de distribution de programmes d'application conformes aux interfaces de programmation des plateformes pour lesquels ils ont été écrits ou aux interfaces de programmation IBM. Ces exemples de programmes n'ont pas été rigoureusement testés dans toutes les conditions. Par conséquent, IBM ne peut garantir expressément ou implicitement la fiabilité, la maintenabilité ou le fonctionnement de ces programmes.

Si vous visualisez ces informations en ligne, il se peut que les photographies et illustrations en couleur n'apparaissent pas à l'écran.

Documentation sur l'interface de programmation

Les informations d'interface de programmation, si elles sont fournies, sont destinées à vous aider à créer un logiciel d'application à utiliser avec ce programme.

Ce manuel contient des informations sur les interfaces de programmation prévues qui permettent au client d'écrire des programmes pour obtenir les services de WebSphere MQ.

Toutefois, lesdites informations peuvent également contenir des données de diagnostic, de modification et d'optimisation. Ces données vous permettent de déboguer votre application.

Important : N'utilisez pas ces informations de diagnostic, de modification et d'optimisation comme interface de programmation car elles sont susceptibles d'être modifiées.

Marques

IBM, le logo IBM , ibm.com, sont des marques d' IBM Corporation dans de nombreux pays. La liste actualisée de toutes les marques d' IBM est disponible sur la page Web "Copyright and trademark

information"www.ibm.com/legal/copytrade.shtml. Les autres noms de produits et de services peuvent être des marques d'IBM ou d’autres sociétés.

Microsoft et Windows sont des marques de Microsoft Corporation aux Etats-Unis et/ou dans d'autres pays.

UNIX est une marque de The Open Group aux Etats-Unis et dans certains autres pays.

Linux est une marque de Linus Torvalds aux Etats-Unis et/ou dans certains autres pays.

Ce produit inclut des logiciels développés par le projet Eclipse (https://www.eclipse.org/).

Java ainsi que tous les logos et toutes les marques incluant Java sont des marques d'Oracle et/ou de ses sociétés affiliées.

Référence :

(1P) P/N: