9.2

IBM MQ nei contenitori

Nota

Prima di utilizzare queste informazioni e il prodotto che supportano, leggere le informazioni in "Informazioni particolari" a pagina 161.

Questa edizione si applica alla versione 9 release 2 di IBM® MQ e a tutte le successive release e modifiche se non diversamente indicato nelle nuove edizioni.

Quando si inviano informazioni a IBM, si concede a IBM un diritto non esclusivo di utilizzare o distribuire le informazioni in qualsiasi modo ritenga appropriato senza incorrere in alcun obbligo verso l'utente.

Indice

Multi IBM MQ nei contenitori e IBM Cloud Pak for Integration

I contenitori ti consentono di impacchettare un gestore code IBM MQ o un'applicazione client IBM MQ , con tutte le sue dipendenze, in un'unità standardizzata per lo sviluppo software.

È possibile eseguire IBM MQ utilizzando IBM MQ Operator su Red Hat® OpenShift®. Questa operazione può essere eseguita utilizzando IBM Cloud Pak for Integration, IBM MQ Advanced o IBM MQ Advanced for Developers.

Puoi anche eseguire IBM MQ in un contenitore che crei da solo.

MQ Adv. CD Per ulteriori informazioni su IBM MQ Operator, consultare i seguenti link.

Multi Pianificazione per IBM MQ nei contenitori

Quando si pianifica IBM MQ nei contenitori, considerare il supporto fornito da IBM MQ per varie opzioni architetturali, ad esempio come viene gestita l'alta disponibilità e come proteggere i gestori code.

Informazioni su questa attività

Prima di pianificare il tuo IBM MQ nell'architettura dei contenitori, dovresti familiarizzare con i concetti IBM MQ di base (vedi IBM MQ Technical overview) e con i concetti di base di Kubernetes/Red Hat OpenShift (vedi Red Hat OpenShift Container Platform architecture).

Procedura

- “Scelta della modalità di utilizzo di IBM MQ nei contenitori” a pagina 5.
- “Supporto per IBM MQ Operator” a pagina 6.
- “Supporto per la creazione di immagini del contenitore del gestore code IBM MQ” a pagina 13.
- “Considerazioni sulla memoria per IBM MQ Operator” a pagina 11.
- “Alta disponibilità per IBM MQ nei contenitori” a pagina 16.
- “Ripristino di emergenza per IBM MQ nei contenitori” a pagina 18.
- “Autenticazione e autorizzazione utente per IBM MQ nei contenitori” a pagina 18.

Scelta della modalità di utilizzo di IBM MQ nei contenitori

Ci sono diverse opzioni per l'utilizzo di IBM MQ nei contenitori: puoi scegliere di utilizzare IBM MQ Operator, che utilizza le immagini del contenitore preconfezionate, oppure puoi creare le tue immagini e il tuo codice di distribuzione.

Utilizzo di IBM MQ Operator

OpenShift CP4I

Se si sta pianificando la distribuzione su Red Hat OpenShift Container Platform, probabilmente si desidera utilizzare IBM MQ Operator.

IBM MQ Operator aggiunge una nuova risorsa personalizzata `QueueManager` a Red Hat OpenShift Container Platform. L'operatore controlla le nuove definizioni del gestore code e le trasforma in risorse di basso livello necessarie, come le risorse `StatefulSet` e `Service` . Nel caso della HA nativa, l'operatore può anche eseguire l'aggiornamento progressivo complesso delle istanze del gestore code. Consultare “Considerazioni sull'esecuzione di un aggiornamento continuo di un gestore code HA nativo” a pagina 155

Alcune funzioni IBM MQ non sono supportate quando si utilizza IBM MQ Operator. Sarà necessario creare le proprie immagini e grafici se si desidera effettuare una delle seguenti operazioni:

- Utilizzare le API REST per la gestione o la messaggistica
- Utilizzare uno dei seguenti componenti di MQ :
  - Managed File Transfer Agent e relative risorse. Tuttavia, è possibile utilizzare IBM MQ Operator per fornire uno o più gestori code Coordinazione, Comando o Agent.
  - AMQP
  - IBM MQ Bridge to Salesforce
  - IBM MQ Bridge to blockchain (non supportato in contenitori)
  - IBM MQ Telemetry Transport (MQTT).
- Personalizzare le opzioni utilizzate con **crtmqm**, **strmqm** e **endmqm**, come la configurazione delle pagine del file di log. La maggior parte delle opzioni può essere configurata utilizzando un file INI.

Tieni presente che IBM MQ Operator e i contenitori si stanno evolvendo rapidamente e pertanto non sono supportati nelle release di Long Term Support .

IBM MQ Operator include sia le immagini del contenitore preintegrate, sia il codice di distribuzione per l'esecuzione su Red Hat OpenShift Container Platform. Il IBM MQ Operator può essere utilizzato per distribuire l'immagine del contenitore IBM MQ fornita o un'immagine del contenitore sovrapposta, ma non può essere utilizzata per distribuire le immagini del contenitore MQ personalizzate.

Creazione di immagini e codice di distribuzione personalizzati

Questa è la soluzione del contenitore più flessibile, ma ti richiede di avere forti capacità nella configurazione dei contenitori e di "possedere" il contenitore risultante. Se non hai intenzione di utilizzare Red Hat OpenShift Container Platform, dovrai creare le tue immagini e il tuo codice di distribuzione.

Sono disponibili esempi per la creazione di immagini personalizzate. Consultare “Creazione del tuo contenitore e del tuo codice di distribuzione IBM MQ” a pagina 147.

Concetti correlati

“Supporto per IBM MQ Operator” a pagina 6
IBM MQ Operator è supportato solo quando viene distribuito su Red Hat OpenShift Container Platform.

“Supporto per la creazione di immagini del contenitore del gestore code IBM MQ” a pagina 13
IBM MQ fornisce il codice per creare un contenitore del gestore code IBM MQ su GitHub. Si basa sul processo che IBM utilizza per creare il proprio contenitore supportato e puoi utilizzare questo repository GitHub per semplificare e accelerare la creazione delle tue immagini del contenitore.

Supporto per IBM MQ Operator

IBM MQ Operator è supportato solo quando viene distribuito su Red Hat OpenShift Container Platform.

IBM MQ Operator utilizza immagini basate sulle release IBM MQ Continuous Delivery (CD), sebbene una release EUS (Extended Update Support) sia disponibile con IBM Cloud Pak for Integration. I rilasci CD sono supportati per un massimo di un anno o per due rilasci CD, a seconda di quale sia il periodo più lungo. Le release Long Term Support di IBM MQ non sono disponibili tramite IBM MQ Operator. IBM Cloud Pak for Integration 2020.4.1 è una release EUS (Extended Update Support), supportata per 18 mesi, se si utilizza una versione di IBM MQ contrassegnata come `-eus`. Altrimenti, IBM MQ 9.2 viene considerato una release Continuous Delivery con IBM MQ Operator.

IBM MQ Operator utilizza immagini contenitore che forniscono un'installazione di IBM MQ su un UBI (Red Hat Universal Base Image), che include le librerie e i programmi di utilità Linux® chiave utilizzati da IBM MQ. L'UBI è supportato da Red Hat quando viene eseguito su Red Hat OpenShift.

IBM MQ Operator è supportato sulle architetture `amd64e` `s390x` (z/Linux).

Concetti correlati

“Supporto per la creazione di immagini del contenitore del gestore code IBM MQ” a pagina 13
IBM MQ fornisce il codice per creare un contenitore del gestore code IBM MQ su GitHub. Si basa sul processo che IBM utilizza per creare il proprio contenitore supportato e puoi utilizzare questo repository GitHub per semplificare e accelerare la creazione delle tue immagini del contenitore.

Supporto versione per IBM MQ Operator

Un'associazione tra le versioni supportate di IBM MQ, Red Hat OpenShift Container Platform e IBM Cloud Pak for Integration.

- “Versioni IBM MQ disponibili” a pagina 7
- “Versioni Red Hat OpenShift Container Platform compatibili” a pagina 8
- “IBM Cloud Pak for Integration versioni” a pagina 8
- “Versioni IBM MQ disponibili in operatori meno recenti” a pagina 8
- “Versioni Red Hat OpenShift Container Platform compatibili per gli operatori meno recenti” a pagina 9

Versioni IBM MQ disponibili

Canale operatore	Versione dell'operatore	IBM MQ versioni							
		9.1.5	CD 9.2.0	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	⚠	⚠	→	⚠	⚠	●	●	●

Chiave:

●
Supporto Continuous Delivery disponibile

▯
Extended Update Support disponibile

→
Disponibile solo durante la migrazione dall'operando Extended Update Support a un operando Continuous Delivery .

⚠
Obsoleto. Poiché le release IBM MQ non sono più supportate, possono essere ancora configurabili nell'operatore, ma non sono più idonee per il supporto e possono essere rimosse nelle release future.

Consultare “Cronologia delle release per IBM MQ Operator” a pagina 20 per i dettagli completi di ciascuna versione, incluse le funzioni dettagliate, le modifiche e le correzioni in ogni versione.

Versioni Red Hat OpenShift Container Platform compatibili

Canale operatore	Versione dell'operatore	Red Hat OpenShift Container Platform versioni[1]				
		4.6	4.7[2]	4.8	4.9	4.10
v1.6	1.6	●	●	●	●	●
v1.7	1.7	●	●	●	●	●
v1.8	1.8	●	●	●	●	●

Chiave:

● Supporto Continuous Delivery disponibile

▯ Extended Update Support disponibile

IBM Cloud Pak for Integration versioni

IBM MQ Operator 1.8.x è supportato per l'utilizzo come parte di IBM Cloud Pak for Integration versione 2021.4.1o indipendentemente.

IBM MQ Operator 1.7.x è supportato per l'utilizzo come parte di IBM Cloud Pak for Integration versione 2021.4.1o indipendentemente.

IBM MQ Operator 1.6.x è supportato per l'utilizzo come parte di IBM Cloud Pak for Integration versione 2021.2.1, 2021.3.1o indipendentemente.

IBM MQ Operator 1.5.x non è più supportato.

IBM MQ Operator 1.4.x non è più supportato.

IBM MQ Operator 1.3.x non è più supportato.

IBM MQ Operator 1.2.x non è più supportato.

Le IBM MQ Operator 1.1.x e 1.0.x non sono più supportate.

Versioni IBM MQ disponibili in operatori meno recenti

La seguente tabella si applica alle versioni di IBM MQ Operator che hanno ora raggiunto la "fine del ciclo di vita".

Canale operatore	Versione dell'operatore	IBM MQ versioni							
		9.1.5	CD 9.2.0	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	⚠	⚠						
v1.3-eus	1.3	⚠	⚠	⚠					

[1] Le versioni di Red Hat OpenShift Container Platform sono soggette alle proprie date di supporto. Per ulteriori informazioni, consultare Red Hat OpenShift Container Platform Politica del ciclo di vita .

[2] IBM MQ Operator dipende da IBM Cloud Pak foundational services. Se si desidera utilizzare Red Hat OpenShift Container Platform 4.7, è necessario aggiornare prima la versione di IBM Cloud Pak foundational services.

Canale operatore	Versione dell'operatore	IBM MQ versioni							
		9.1.5	CD 9.2.0	9.2.0 EUS	9.2.1	9.2.2	9.2.3	9.2.4	9.2.5
v1.4	1.4	⚠	⚠	→	⚠				
v1.5	1.5	⚠	⚠	→	⚠	⚠			

Chiave:

→
Disponibile solo durante la migrazione dall'operando Extended Update Support a un operando Continuous Delivery .

⚠
Obsoleto. Poiché le release di IBM MQ non sono più supportate, possono essere ancora configurabili in IBM MQ Operator, ma non sono più idonee per il supporto.

Consultare “Cronologia delle release per IBM MQ Operator” a pagina 20 per i dettagli completi di ciascuna versione, incluse le funzioni dettagliate, le modifiche e le correzioni in ogni versione.

Versioni Red Hat OpenShift Container Platform compatibili per gli operatori meno recenti

La seguente tabella si applica alle versioni di IBM MQ Operator che hanno ora raggiunto la "fine del ciclo di vita".

Canale operatore	Versione dell'operatore	Red Hat OpenShift Container Platform versioni[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			⚠	⚠	⚠	⚠	⚠

Chiave:

→
Disponibile solo durante la migrazione dall'operando Extended Update Support a un operando Continuous Delivery .

[3] Le versioni di Red Hat OpenShift Container Platform sono soggette alle proprie date di supporto. Per ulteriori informazioni, consultare Red Hat OpenShift Container Platform Politica del ciclo di vita .

[4] Red Hat OpenShift Container Platform 4.4 ha raggiunto la "fine del ciclo di vita". Per ulteriori informazioni, consultare Red Hat OpenShift Container Platform Politica del ciclo di vita .

[5] Red Hat OpenShift Container Platform 4.5 ha raggiunto la "fine del ciclo di vita". Per ulteriori informazioni, consultare Red Hat OpenShift Container Platform Politica del ciclo di vita .

[6] IBM MQ Operator dipende da IBM Cloud Pak foundational services. Se si desidera utilizzare Red Hat OpenShift Container Platform 4.7, è necessario aggiornare prima la versione di IBM Cloud Pak foundational services.

⚠

La versione di IBM MQ Operator ha raggiunto la "fine del ciclo di vita", ma era precedentemente disponibile su questa versione di Red Hat OpenShift Container Platform

OpenShift CP4I Dipendenze per IBM MQ Operator

IBM MQ Operator dipende dall'operatore IBM Cloud Pak foundational services , che installa anche l'operatore IBM Operand Deployment Lifecycle Manager (ODLM). Questi operatori verranno installati automaticamente quando si installa IBM MQ Operator. Questi operatori dipendenti hanno una piccola quantità di CPU e memoria e vengono utilizzati per distribuire ulteriori risorse in alcune circostanze.

Quando crei un `QueueManager`, IBM MQ Operator creerà un `OperandRequest` per i servizi aggiuntivi di cui ha bisogno. Il `OperandRequest` viene soddisfatto dall'operatore ODLM e, se necessario, installerà e creerà un'istanza dei servizi richiesti. I servizi richiesti sono determinati in base all'accordo di licenza accettato durante la distribuzione del gestore code e in base ai componenti del gestore code richiesti.

- Se si sceglie una licenza IBM MQ Advanced o IBM MQ Advanced for Developers , non sono richiesti ulteriori servizi. Ad esempio, nel seguente caso, IBM Cloud Pak foundational services non viene utilizzato:

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

- Se si sceglie una licenza IBM Cloud Pak for Integration e si sceglie di abilitare il server web, IBM MQ Operator creerà anche un'istanza di IBM Identity and Access Management (IAM) Operator, per abilitare l'SSO (single sign - on). L'operatore IAM sarà già disponibile se è stato installato l'operatore IBM Cloud Pak for Integration . Ad esempio:

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

  Tuttavia, se si disabilita il server Web, non viene richiesto alcun IBM Cloud Pak foundational services . Ad esempio:

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

Le versioni precedenti di IBM MQ Operator richiedevano sempre l'installazione di IBM Licensing Operator (e relative dipendenze), per tenere traccia dell'utilizzo della licenza. A partire da IBM MQ Operator 1.5 in poi, il servizio di licenza non è richiesto ed è necessario richiederlo separatamente.

IBM MQ Operator richiede 1 core CPU e 1 GB di memoria. Per una suddivisione dettagliata dei requisiti hardware e software per gli operatori dipendenti, consultare Requisiti hardware e raccomandazioni per i servizi fondamentali.

È possibile scegliere la quantità di CPU e memoria utilizzata dai gestori code. Per ulteriori informazioni, consultare “.spec.queueManager.resources” a pagina 135.

Riferimenti correlati

“Riferimento per la licenza per mq.ibm.com/v1beta1” a pagina 126

OpenShift CP4I Autorizzazioni nell'ambito del cluster richieste da IBM MQ Operator

IBM MQ Operator richiede autorizzazioni nell'ambito del cluster per gestire i webhook di ammissione e gli esempi e per leggere le informazioni sulla classe di archiviazione e sulla versione cluster.

IBM MQ Operator richiede le seguenti autorizzazioni nell'ambito del cluster:

- Autorizzazione a gestire i webhook di ammissione. Ciò consente di creare, richiamare e aggiornare specifici webhook utilizzati nel processo di creazione e gestione dei contenitori forniti dall'operatore.
  - Gruppi API: **admissionregistration.k8s.io**
  - Risorse: **validatingwebhookconfigurations**
  - verbs: **create, get, update**
- Autorizzazione per creare e gestire le risorse utilizzate nella console Red Hat OpenShift per fornire esempi e frammenti quando si creano risorse personalizzate.
  - Gruppi API: **console.openshift.io**
  - Risorse: **consoleyamlsamples**
  - verbs: **create, get, update, delete**
- Autorizzazione a leggere la versione del cluster. Ciò consente all'operatore di eseguire il feed back di eventuali problemi con l'ambiente cluster.
  - Gruppi API: **config.openshift.io**
  - Risorse: **clusterversions**
  - verbs: **get, list, watch**
- Autorizzazione a leggere le classi di memoria sul cluster. Ciò consente all'operatore di eseguire il feed di eventuali problemi con le classi di archiviazione selezionate nei contenitori.
  - Gruppi API: **storage.k8s.io**
  - Risorse: **storageclasses**
  - verbs: **get, list**

OpenShift CP4I Kubernetes Considerazioni sulla memoria per IBM MQ Operator

IBM MQ Operator viene eseguito in due modalità di memoria:

- L' **archiviazione temporanea** viene utilizzata quando tutte le informazioni sullo stato del contenitore possono essere eliminate quando il contenitore viene riavviato. Viene comunemente utilizzato quando gli ambienti vengono creati per la dimostrazione o quando si sviluppano con gestori code autonomi.
- L' **archiviazione persistente** è la configurazione comune per IBM MQ e garantisce che se il contenitore viene riavviato, la configurazione esistente, i log e i messaggi persistenti sono disponibili nel contenitore riavviato.

IBM MQ Operator fornisce la possibilità di personalizzare le caratteristiche di archiviazione che possono differire notevolmente a seconda dell'ambiente e della modalità di archiviazione desiderata.

Archiviazione effimera

IBM MQ è un'applicazione con stato e conserva questo stato nella memoria per il recupero in caso di riavvio. Se si utilizza la memoria temporanea, tutte le informazioni sullo stato per il gestore code vengono perse al riavvio. Questo include:

- Tutti i messaggi

- Tutti i gestori code allo stato di comunicazione del gestore code (numeri di sequenza dei messaggi del canale)
- L'identità cluster MQ del gestore code
- Stato di tutte le transazioni
- Configurazione di tutti i gestori code
- Tutti i dati diagnostici locali

Per questo motivo è necessario considerare se l'archiviazione effimera è un approccio adatto per uno scenario di produzione, test o sviluppo. Ad esempio, dove tutti i messaggi sono noti come non persistenti e il gestore code non è un membro di un cluster MQ . Oltre all'eliminazione di tutto lo stato di messaggistica al riavvio, viene eliminata anche la configurazione del gestore code. Per abilitare un contenitore completamente effimero la configurazione IBM MQ deve essere aggiunta all'immagine del contenitore stessa (per ulteriori informazioni, consultare “Creazione di un'immagine con file MQSC e INI personalizzati, utilizzando la CLI Red Hat OpenShift” a pagina 112). Se questo non viene completato, IBM MQ dovrà essere configurato ogni volta che il contenitore viene riavviato.

Ad esempio, per configurare IBM MQ con memoria effimera, il tipo di archiviazione di `QueueManager` deve includere quanto segue:

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

Archiviazione permanente

IBM MQ viene normalmente eseguito con la memoria persistente per garantire che il gestore code conservi i messaggi persistenti e la configurazione dopo un riavvio. Pertanto, questo è il comportamento predefinito. A causa dei vari provider di memoria e delle diverse funzionalità di ciascun supporto, ciò spesso significa che è richiesta la personalizzazione della configurazione. L'esempio riportato di seguito descrive i campi comuni che personalizzano la configurazione di archiviazione di MQ nell'API v1beta1 :

- spec.queueManager.availability controlla la modalità di disponibilità. Se si utilizza `SingleInstance` , è necessaria solo l'archiviazione `ReadWriteOnce` , mentre `multiInstance` richiede una classe di archiviazione che supporta `ReadWriteMany` con le caratteristiche di blocco file corrette. IBM MQ fornisce un' istruzione di supporto e un' istruzione di test. La modalità di disponibilità influenza anche il layout del volume persistente. Per ulteriori informazioni, consultare “Alta disponibilità per IBM MQ nei contenitori” a pagina 16
- spec.queueManager.storage controlla le singole impostazioni di memoria. Un gestore code può essere configurato per utilizzare tra uno e quattro volumi persistenti

Il seguente esempio mostra un frammento di una configurazione semplice utilizzando un gestore code a istanza singola:

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

Il seguente esempio mostra un frammento di configurazione di un gestore code a più istanze, con una classe di memoria non predefinita e con l'archiviazione file che richiede gruppi supplementari:

```
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]
```

Nota: È inoltre possibile configurare gruppi supplementari con gestori code a istanza singola.

Nota: Non sono richiesti file system condivisi se si utilizza la HA nativa (consultare “Alta disponibilità per IBM MQ nei contenitori” a pagina 16). In particolare, non utilizzare NFSv3.

Linux Supporto per la creazione di immagini del contenitore del gestore code IBM MQ

IBM MQ fornisce il codice per creare un contenitore del gestore code IBM MQ su GitHub. Si basa sul processo che IBM utilizza per creare il proprio contenitore supportato e puoi utilizzare questo repository GitHub per semplificare e accelerare la creazione delle tue immagini del contenitore.

Il codice viene fornito nel repository GitHub del contenitore mq: https://github.com/ibm-messaging/mq-container. Viene fornito con una licenza Apache 2.0 , con il supporto fornito dalla community.

Il repository non utilizza i package rpm standard di Linux ; utilizza il package compresso per le distribuzioni del contenitore. Il vantaggio di questo approccio è che è possibile eseguire in ambienti contenitore più sicuri senza la necessità di autorizzazioni sottoposte a escalation. Tuttavia, ciò influisce sulle opzioni di sicurezza disponibili, poiché IBM MQ utilizza tradizionalmente le autorizzazioni di escalation per l'autenticazione basata sul sistema operativo. Per una distribuzione del contenitore, l'utilizzo dell'autenticazione basata sul sistema operativo non è di norma una buona pratica; è invece possibile utilizzare l'autenticazione TLS o LDAP reciproca. Con IBM MQ Advanced for Developers, puoi anche utilizzare l'autenticazione basata su file, consentendo ai tuoi utenti di iniziare rapidamente.

Il gestore code di dati replicati (RDQM) non è supportato in un ambiente contenitore. È possibile ottenere funzionalità simili a RDQM utilizzando “HA nativa” a pagina 92.

Concetti correlati

“Supporto per IBM MQ Operator” a pagina 6
IBM MQ Operator è supportato solo quando viene distribuito su Red Hat OpenShift Container Platform.

Immagini di non installazione IBM MQ

Linux Annotazioni di licenza durante la creazione della propria immagine del contenitore IBM MQ

Le annotazioni di licenza ti permettono di tenere traccia dell'utilizzo in base ai limiti definiti sul contenitore, piuttosto che sulla macchina sottostante. Configura i tuoi client per distribuire il contenitore con annotazioni specifiche che IBM License Service utilizza per tracciare l'utilizzo.

Quando si distribuisce un'immagine del contenitore IBM MQ auto - costruito, ci sono due approcci comuni alla licenza:

- Licenza dell'intera macchina che esegue il contenitore.
- Licenza del contenitore in base ai limiti associati.

Entrambe le opzioni sono disponibili per i client e ulteriori dettagli possono essere trovati nella pagina IBM Container Licenses in Passport Advantage.

Se il contenitore IBM MQ deve essere concesso in licenza in base ai limiti del contenitore, IBM License Service deve essere installato per tenere traccia dell'utilizzo. Ulteriori informazioni relative agli ambienti supportati e alle istruzioni di installazione sono disponibili nella pagina ibm - licensing - operator su GitHub.

IBM License Service è installato sul cluster Kubernetes in cui è distribuito il contenitore IBM MQ e le annotazioni del pod vengono utilizzate per tracciare l'utilizzo. Pertanto i client devono distribuire il pod

con annotazioni specifiche che IBM License Service utilizza. In base alla tua titolarità e alle funzionalità distribuite nel contenitore, utilizza una o più delle seguenti annotazioni:

IBM MQ Contenitore avanzato

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

Contenitore 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"
```

IBM MQ Contenitore di base

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

Contenitore 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"
```

Contenitore IBM MQ Advanced for Developers

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

IBM MQ Contenitore avanzato con titolarità CP4I (Produzione)

```
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"
```

Contenitore IBM MQ Advanced High Availability Replica con titolarità CP4I (Produzione)

```
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"
```

IBM MQ Contenitore avanzato con titolarità CP4I (non di produzione)

```
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"
```

Contenitore IBM MQ Advanced High Availability Replica con titolarità CP4I (Non di produzione)

```
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"
```

IBM MQ Base con titolarità CP4I (Produzione)

```
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"
```

IBM MQ Base High Availability Replica con titolarità CP4I (Produzione)

```
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"
```

IBM MQ Base con una titolarità CP4I (Non di produzione)

```
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"
```

IBM MQ Base High Availability Replica con titolarità CP4I (Non di produzione)

```
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 Alta disponibilità per IBM MQ nei contenitori

Ci sono tre scelte per l'alta disponibilità con IBM MQ Operator: **gestore code HA nativa** (che ha una replica attiva e due repliche in standby), **gestore code a più istanze** (che è una coppia attivo - standby, che utilizza un file system condiviso in rete) o **Gestore code resiliente singolo** (che offre un approccio semplice per l'HA che utilizza la memoria in rete). Gli ultimi due si basano sul sistema di file per assicurare la disponibilità dei dati recuperabili, tuttavia Native HA non lo fa. Pertanto, quando non si utilizza la HA nativa, la disponibilità del file system è critica per la disponibilità del gestore code. Quando il recupero dei dati è importante, il file system deve garantire la ridondanza attraverso la replica.

Dovresti considerare separatamente la disponibilità del **messaggio** e del **servizio** . Con IBM MQ for Multiplatforms, un messaggio viene memorizzato esattamente su un gestore code. Quindi, se il gestore code diventa non disponibile, si perde temporaneamente l'accesso ai messaggi in esso contenuti. Per ottenere un'elevata disponibilità di `messaggi` , è necessario essere in grado di ripristinare un gestore code il più rapidamente possibile. Puoi ottenere la disponibilità del `servizio` disponendo di più istanze di code per le applicazioni client da utilizzare, ad esempio utilizzando un cluster uniforme IBM MQ .

Un gestore code può essere pensato in due parti: i dati memorizzati sul disco e i processi in esecuzione che consentono l'accesso ai dati. Qualsiasi gestore code può essere spostato in un nodo Kubernetes diverso, purché conservi gli stessi dati (forniti dai volumi persistenti Kubernetes) ed è ancora indirizzabile nella rete dalle applicazioni client. In Kubernetes, un servizio è utilizzato per fornire un'identità di rete congruente.

IBM MQ si basa sulla disponibilità dei dati sui volumi persistenti. Pertanto, la disponibilità della memoria che fornisce i volumi persistenti è fondamentale per la disponibilità del gestore code, perché IBM MQ non può essere più disponibile della memoria che sta utilizzando. Se si desidera tollerare un'interruzione di un'intera zona di disponibilità, è necessario utilizzare un provider di volumi che replichi le scritture disco in un'altra zona.

Gestore code HA nativo

CP4I V 9.2.3

I gestori code della HA nativa sono disponibili a partire da IBM Cloud Pak for Integration 2021.2.1, utilizzando IBM MQ Operator 1.6 o versioni successive, con IBM MQ 9.2.3 o versioni successive.

I gestori code della HA nativa coinvolgono un **attivo** e due pod di **replica** Kubernetes , che vengono eseguiti come parte di un StatefulSet Kubernetes con esattamente tre repliche ciascuna con la propria

serie di Kubernetes Volumi persistenti. I requisiti IBM MQ per i file system condivisi si applicano anche quando si utilizza un gestore code HA nativo (tranne per il blocco basato sul lease), ma non è necessario utilizzare un file system condiviso. È possibile utilizzare l'archiviazione blocchi, con un file system adatto in cima. Ad esempio, *xfs* o *ext4*. I tempi di recupero per un gestore code HA nativo sono controllati dai seguenti fattori:

1. Il tempo necessario alle istanze di replica per rilevare che l'istanza attiva ha avuto esito negativo. Questo è configurabile.
2. Il tempo impiegato dal probe Pod di Kubernetes per rilevare che il contenitore pronto è stato modificato e reindirizzare il traffico di rete. Questo è configurabile.
3. Il tempo necessario ai client IBM MQ per riconnettersi.

Per ulteriori informazioni, consultare “HA nativa” a pagina 92

gestore code a più istanze

I gestori code a più istanze coinvolgono un pod **attivo** e un pod **standby** Kubernetes , che vengono eseguiti come parte di un Kubernetes Stateful Set con esattamente due repliche e una serie di volumi persistenti Kubernetes . I dati e i log delle transazioni del gestore code sono conservati su due volumi permanenti, utilizzando un filesystem condiviso.

I gestori code a più istanze richiedono che i pod **attivi** e **standby** abbiano accesso simultaneo al volume persistente. Per configurare ciò, utilizzare Kubernetes Volumi persistenti con **`access mode`** impostato su `ReadWriteMany`. I volumi devono inoltre soddisfare i requisiti di IBM MQ per i file system condivisi, poiché IBM MQ si basa sul rilascio automatico dei blocchi file per istigare un failover del gestore code. IBM MQ produce un elenco di file system verificati.

I tempi di recupero per un gestore code a più istanze sono controllati dai seguenti fattori:

1. Il tempo impiegato dopo che si è verificato un malfunzionamento per il file system condiviso per rilasciare i blocchi originariamente presi dall'istanza attiva.
2. Il tempo impiegato dall'istanza standby per acquisire i blocchi e quindi avviarli.
3. Il tempo impiegato dal probe Pod di Kubernetes per rilevare che il contenitore pronto è stato modificato e reindirizzare il traffico di rete. Questo è configurabile.
4. Il tempo impiegato dai client IBM MQ per riconnettersi.

Singolo gestore code resiliente

Un singolo gestore code resiliente è una singola istanza di un gestore code in esecuzione in un singolo pod Kubernetes , dove Kubernetes monitorizza il gestore code e sostituisce il pod come necessario.

I requisiti di IBM MQ per i file system condivisi si applicano anche quando si utilizza un singolo gestore code resiliente (ad eccezione del blocco basato sul lease), ma non è necessario utilizzare un file system condiviso. È possibile utilizzare l'archiviazione blocchi, con un file system adatto in cima. Ad esempio, *xfs* o *ext4*.

I tempi di recupero per un singolo gestore code resiliente sono controllati dai seguenti fattori:

1. Il tempo impiegato per l'esecuzione del probe di attività e il numero di errori tollerati. Questo è configurabile.
2. Il tempo impiegato dallo scheduler Kubernetes per ripianificare il pod non riuscito su un nuovo nodo.
3. Quanto tempo ci vuole per scaricare l'immagine del contenitore sul nuovo Nodo. Se si utilizza un valore **`imagePullPolicy`** di `IfNotPresent`, l'immagine potrebbe essere già disponibile su tale Nodo.
4. Il tempo impiegato per l'avvio della nuova istanza del gestore code.
5. Il tempo impiegato dal probe di disponibilità del pod Kubernetes per rilevare che il contenitore è pronto. Questo è configurabile.

6. Il tempo impiegato dai client IBM MQ per riconnettersi.

Importante:

Sebbene il singolo modello di gestore code resiliente offra alcuni vantaggi, è necessario comprendere se è possibile raggiungere i propri obiettivi di disponibilità con le limitazioni relative agli errori del nodo.

In Kubernetes, un pod malfunzionante viene generalmente ripristinato rapidamente, ma l'errore di un intero nodo viene gestito in modo diverso. Quando si utilizza un carico di lavoro con stato come IBM MQ con uno Kubernetes StatefulSet, se un nodo master Kubernetes perde il contatto con un nodo di lavoro, non può determinare se il nodo ha avuto esito negativo o se ha semplicemente perso la connettività di rete. Pertanto, Kubernetes non esegue **alcuna azione** in questo caso finché non si verifica uno dei seguenti eventi:

1. Il nodo viene ripristinato in uno stato in cui il nodo master Kubernetes può comunicare con esso.
2. Viene eseguita un'azione amministrativa per eliminare esplicitamente il pod sul nodo master Kubernetes . Ciò non arresta necessariamente l'esecuzione del pod, ma lo elimina semplicemente dall'archivio Kubernetes . Questa azione amministrativa deve quindi essere intrapresa con molta attenzione.

Attività correlate

“Configurazione dell'alta disponibilità per i gestori code utilizzando IBM MQ Operator” a pagina 92

Riferimenti correlati

Configurazioni HA (High Availability)

OpenShift CP4I Kubernetes Ripristino di emergenza per IBM MQ nei contenitori

Devi considerare a quale tipo di disastro ti stai preparando. Negli ambienti cloud, l'utilizzo delle zone di disponibilità fornisce un certo livello di tolleranza per i disastri e sono molto più facili da usare. Se hai un numero dispari di data center (per quorum) e un link di rete a bassa latenza, puoi potenzialmente eseguire un singolo cluster Red Hat OpenShift Container Platform o Kubernetes con più zone di disponibilità, ognuna in un'ubicazione fisica separata. Questo argomento illustra le considerazioni per il ripristino di emergenza in cui questi criteri non possono essere soddisfatti, ovvero un numero pari di data center o un link di rete ad alta latenza.

Per il ripristino di emergenza, è necessario considerare quanto segue:

- Replica dei dati IBM MQ (conservati in una o più risorse `PersistentVolume`) nell'ubicazione di ripristino di emergenza
- Ricreazione del gestore code utilizzando i dati replicati
- L'ID di rete del gestore code visibile alle applicazioni client IBM MQ e ad altri gestori code. Questo ID potrebbe essere una voce DNS, ad esempio.

I dati persistenti devono essere replicati, in modo sincrono o asincrono, sul sito di ripristino di emergenza. Ciò è in genere specifico del provider di memoria, ma può essere eseguito anche utilizzando un `VolumeSnapshot`. Consultare Istantanee volume CSI per ulteriori informazioni sulle istantanee volume.

Quando si esegue il ripristino da un'emergenza, sarà necessario ricreare l'istanza del gestore code sul nuovo cluster Kubernetes , utilizzando i dati replicati. Se stai utilizzando IBM MQ Operator, avrai bisogno dello YAML `QueueManager` e dello YAML per altre risorse di supporto come `ConfigMap` o `Secret`.

Informazioni correlate

ha_for_ctr.dita

Autenticazione e autorizzazione utente per IBM MQ nei contenitori

IBM MQ può essere configurato per utilizzare utenti e gruppi LDAP. In alternativa, puoi utilizzare utenti e gruppi del sistema operativo locale all'interno dell'immagine del contenitore. IBM MQ Operator non consente l'utente di utenti e gruppi del sistema operativo a causa di problemi di sicurezza.

In un ambiente containerizzato a più tenant, i vincoli di sicurezza vengono di norma messi in atto per impedire potenziali problemi di sicurezza, ad esempio:

- **Come impedire l'utilizzo dell'utente "root" in un contenitore**
- **forzatura dell'uso di un UID casuale**. Ad esempio, in Red Hat OpenShift Container Platform il valore predefinito `SecurityContextConstraints` (denominato `restricted`) utilizza un ID utente casuale per ogni contenitore.
- **Impedire l'utilizzo dell'escalation di privilegi**. IBM MQ on Linux utilizza l'escalation dei privilegi per controllare le password degli utenti - utilizza un programma "setuid" per diventare l'utente "root" per farlo.

Per garantire la conformità con queste misure di protezione, IBM MQ Operator non consente l'utilizzo di ID che sono definiti nelle librerie del sistema operativo all'interno di un contenitore. Nessun ID utente o gruppo `mqm` definito nel contenitore. Quando si utilizza IBM MQ in IBM Cloud Pak for Integration e Red Hat OpenShift, è necessario configurare il proprio gestore code per utilizzare LDAP per l'autenticazione utente e l'autorizzazione. Per informazioni sulla configurazione di IBM MQ a tale scopo, consultare Autenticazione connessione: repository utente e autorizzazione LDAP

Pianificazione della scalabilità e delle prestazioni per IBM MQ nei contenitori

Nella maggior parte dei casi, la scalabilità e le prestazioni di IBM MQ nei contenitori sono le stesse di IBM MQ for Multiplatforms. Tuttavia, ci sono alcuni limiti aggiuntivi che possono essere imposti dalla piattaforma del contenitore.

Informazioni su questa attività

Quando si pianifica la scalabilità e le prestazioni per IBM MQ nei contenitori, considerare le opzioni riportate di seguito:

Procedura

- **Limitare il numero di thread e processi.**

  IBM MQ utilizza i thread per gestire la simultaneità. In Linux, i thread vengono implementati come processi, in modo che sia possibile incontrare i limiti imposti dalla piattaforma del contenitore o dal sistema operativo, sul numero massimo di processi. In Red Hat OpenShift Container Platform, esiste un limite predefinito di 4096 processi per contenitore (1024 processi fino a OpenShift 4.11). Sebbene ciò sia adeguato per la maggior parte degli scenari, è possibile che ciò influisce sul numero di connessioni client per un gestore code.

  Il limite di processi in Kubernetes può essere configurato dall'amministratore del cluster utilizzando l'impostazione di configurazione kubelet **`podPidsLimit`**. Vedi Limiti e prenotazioni dell'ID processo nella documentazione di Kubernetes . In Red Hat OpenShift Container Platform, puoi anche creare una **`ContainerRuntimeConfig`** risorsa personalizzata per modificare i CRI-O.

  Nella configurazione IBM MQ , è anche possibile impostare il numero massimo di connessioni client per un gestore code. Consultare Limiti del canale di connessione server per l'applicazione dei limiti a un singolo canale di connessione server e l' attributo MAXCHANNELS INI per applicare i limiti all'intero gestore code.
- **Limita numero di volumi.**

  Nei sistemi cloud e contenitore, i volumi di archiviazione collegati alla rete sono comunemente utilizzati. Esistono dei limiti al numero di volumi che possono essere collegati ai nodi Linux . Ad esempio, AWS EC2 limita a non più di 30 volumi per VM. Red Hat OpenShift Container Platform ha un limite simile, come Microsoft Azure e Google Cloud Platform.

  Un gestore code HA nativo richiede un volume per ognuna delle tre istanze e impone la diffusione delle istanze tra i nodi. Tuttavia, è possibile configurare il gestore code in modo che utilizzi tre volumi per istanza (dati del gestore code, log di ripristino e dati persistenti).

- **Utilizza tecniche di scaling IBM MQ .**

  Invece di un numero ridotto di gestori code di grandi dimensioni, può essere utile utilizzare le tecniche di scalabilità IBM MQ come i cluster uniformi IBM MQ per eseguire più gestori code con la stessa configurazione. Ciò ha il vantaggio aggiunto che l'impatto del riavvio di un singolo contenitore (ad esempio, come parte della manutenzione della piattaforma del contenitore) è ridotto.

OpenShift CP4I EUS CD Utilizzo di IBM MQ in IBM Cloud Pak for Integration e Red Hat OpenShift

IBM MQ Operator distribuisce e gestisce IBM MQ come parte di IBM Cloud Pak for Integrationo autonomo su Red Hat OpenShift Container Platform

Procedura

- “Cronologia delle release per IBM MQ Operator” a pagina 20.
- “Migrazione di IBM MQ a IBM Cloud Pak for Integration” a pagina 36.
- “Installazione e disinstallazione di IBM MQ Operator su Red Hat OpenShift” a pagina 59.
- “Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 71.
- “Distribuzione e configurazione dei gestori code utilizzando IBM MQ Operator” a pagina 79.
- “Utilizzo di IBM MQ mediante IBM MQ Operator” a pagina 114.
- “Riferimento API per IBM MQ Operator” a pagina 125.

OpenShift CP4I EUS CD Cronologia delle release per IBM MQ Operator

IBM MQ Operator

IBM MQ Operator 1.8.2

CD

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2021.4.1

Canale operatore
`v1.8`

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 e successive

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.8 e successive (canalev3)

Novità

- Aggiornamento di sicurezza basato su IBM MQ Operator 1.8.0.
- Le vulnerabilità risolte sono descritte in questo Bollettino sulla sicurezza.

IBM MQ Operator 1.8.1

CD

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2021.4.1

Canale operatore
v1.8

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 e successive

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.8 e successive (canalev3)

Novità

- Aggiornamento di sicurezza basato su IBM MQ Operator 1.8.0.
- Le vulnerabilità risolte sono descritte in questo Bollettino sulla sicurezza.

IBM MQ Operator 1.8.0

CD

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2021.4.1

Canale operatore
v1.8

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 e successive

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.8 e successive (canalev3)

Novità

- Aggiunge condizioni di stato per versioni IBM MQ obsolete.

Elementi modificati

- Immagini spostate da Docker Hub a IBM Container Registry.
  - I clienti con regole firewall potrebbero dover modificarle per accedere alle immagini su IBM Container Registry.
  - I clienti Airgap riscontrano un riavvio del nodo durante l'aggiornamento a IBM MQ Operator 1.8.0.
- Versioni obsolete: IBM MQ 9.1.5, CD 9.2.0 , 9.2.1, 9.2.2. Queste versioni potrebbero non essere riconciliate dalle versioni future di IBM MQ Operator.
- Modifiche alla logica della licenza: i Clienti che eseguono l'upgrade a IBM MQ 9.2.5 possono utilizzare solo le licenze specificate per lavorare con IBM MQ 9.2.5. Vedere “Riferimento per la licenza per mq.ibm.com/v1beta1” a pagina 126.
- Le vulnerabilità risolte sono descritte in questo Bollettino sulla sicurezza.

IBM MQ Operator 1.7.0

CD

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2021.4.1

Canale operatore
v1.7

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 e successive

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.8 e successive (canalev3)

Novità

- Aggiunge IBM MQ 9.2.4 come release di fornitura continua

IBM MQ Operator 1.6.0

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2021.2.1

Canale operatore
v1.6

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 e successive

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.7 e successive (canalev3)

Novità

- Aggiunge IBM MQ 9.2.3 come release di fornitura continua (amd64 solo per IBM Cloud Pak for Integration 2021.2.1; amd64 o s390x quando si utilizza una licenza IBM MQ)
- Nuovo tipo di disponibilità per i gestori code: HA nativa. Disponibile per uso di produzione, come parte di IBM Cloud Pak for Integration 2021.2.1.

Elementi modificati

- IBM MQ Operator 1.6 e versioni successive utilizzano IBM Container Registry invece di Docker Hub. Ciò significa che è necessario utilizzare un `CatalogSource` da `icr.io`. Vedere “Installazione e disinstallazione di IBM MQ Operator su Red Hat OpenShift” a pagina 59.
- L'aggiornamento continuo della HA nativa non attende più che una replica sia sincronizzata prima di passare alla replica successiva.
- Risolve i problemi con l'affinità Native HA su OCP 4.7 e versioni successive.
- Risolve il problema quando si utilizzano i certificati firmati CA con la HA nativa.

IBM MQ Operator 1.5.0

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2021.1.1

Canale operatore
v1.5

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 e successive

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.7 e successive (canalev3)

Novità

- Aggiunge IBM MQ 9.2.2 come release di fornitura continua (amd64 solo per IBM Cloud Pak for Integration 2021.1.1; amd64 o s390x quando si utilizza una licenza IBM MQ)
- Nuovo tipo di disponibilità per i gestori code: HA nativa. Disponibile solo a scopo di valutazione, come parte di IBM Cloud Pak for Integration 2021.1.1.
- Integrazione con le metriche Red Hat OpenShift Container Platform Cluster Monitoring for Prometheus , fornendo una risorsa `ServiceMonitor`

Elementi modificati

- L'operatore di licenza IBM non viene pi - creato per impostazione predefinita quando si crea un gestore code
- Gli aggiornamenti ai gestori code a più istanze vengono ora gestiti in un ordine progressivo. Come parte di questa modifica, è stato introdotto un probe di avvio Kubernetes che influisce sui valori utilizzati durante la configurazione del probe di attività. L'analisi di avvio viene avviata immediatamente, quindi attende il corretto avvio del gestore code. Se il probe di avvio passa in qualsiasi momento entro questo periodo di attesa, i probe di disponibilità e di attività vengono avviati. In precedenza, se si aveva un gestore code che era lento ad avviarsi, è possibile che sia stata aumentata l'impostazione `initialDelaySeconds` sul probe di attività. Se è stata fatta questa operazione, è necessario ripristinare `initialDelaySeconds` all'impostazione precedente.
- `CustomResourceDefinition` viene aggiornato da `apiextensions.k8s.io/v1beta1` a `apiextensions.k8s.io/v1`

Problemi e limitazioni noti

- Richiede IBM Cloud Pak foundational services 3.7, che contiene una modifica incompatibile nel componente IAM (Identity and Access Management). Se si dispone di gestori code che utilizzano una licenza IBM Cloud Pak for Integration , dopo questo aggiornamento, sarà richiesto un riavvio del gestore code per accedere alla console Web e verranno visualizzati anche altri errori di accesso alla console Web. È possibile correggere questi errori eseguendo l'upgrade al valore più recente di `.spec.version` per la versione IBM MQ scelta, una volta completato l'aggiornamento dell'operatore.
- L'aggiornamento progressivo non viene avviato automaticamente se si sta aggiornando la versione di MQ . Devi eliminare manualmente i pod.

IBM MQ Operator 1.4.0

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.4.1 (IBM MQ Operator 1.4.0 è una release CD e non è idoneo per Extended Update Support)

Canale operatore
`v1.4`

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 e successive

Novità

- Aggiunge IBM MQ 9.2.1 come release di fornitura continua

- È ora possibile impedire la creazione dell'instradamento del gestore code predefinito impostando `.spec.queueManager.route.enabled` su `false`

Problemi e limitazioni noti

- Quando si aggiorna un `QueueManager` con un tipo di disponibilità `MultiInstance`, entrambi i pod verranno eliminati immediatamente. Entrambi dovrebbero essere riavviati rapidamente da Red Hat OpenShift Container Platform.

IBM MQ Operator 1.3.8 (EUS)

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.4.1

Canale operatore
`v1.3-eus`

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 solo

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.6 (canale`stable-v1`)

Novità

- Aggiunge la nuova versione dell'operando 9.2.0.6-r3-eus.
- Le vulnerabilità risolte sono descritte in questo Bollettino sulla sicurezza.

IBM MQ Operator 1.3.7 (EUS)

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.4.1

Canale operatore
`v1.3-eus`

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 solo

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.6 (canale`stable-v1`)

Novità

- Aggiunge un nuovo operando versione 9.2.0.6-r2-eus.
- Le vulnerabilità risolte sono descritte in questo Bollettino sulla sicurezza.

IBM MQ Operator 1.3.6 (EUS)

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.4.1

Canale operatore
`v1.3-eus`

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 solo

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.6 (canale `stable-v1`)

Novità

- Aggiunge la nuova versione dell'operando 9.2.0.6-r1-eus.
- Le vulnerabilità risolte sono descritte in questo Bollettino sulla sicurezza.

IBM MQ Operator 1.3.5 (EUS)

EUS

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.4.1

Canale operatore
`v1.3-eus`

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 solo

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.6 (canale `stable-v1`)

Novità

- Aggiunge la versione del nuovo operando 9.2.0.5-r3-eus.
- Le vulnerabilità risolte sono descritte in questo Bollettino sulla sicurezza.

IBM MQ Operator 1.3.4 (EUS)

EUS

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.4.1

Canale operatore
`v1.3-eus`

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 solo

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.6 (canale `stable-v1`)

Novità

- Aggiunge una nuova versione dell'operando 9.2.0.5-r2-eus
- Le vulnerabilità risolte sono descritte in questo Bollettino sulla sicurezza.

IBM MQ Operator 1.3.3 (EUS)

EUS

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.4.1

Canale operatore
`v1.3-eus`

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 solo

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.6 (canale `stable-v1`)

Novità

- Aggiunge la nuova versione dell'operando, 9.2.0.5-r1-eus
- Le vulnerabilità risolte sono descritte in questo Bollettino sulla sicurezza.

IBM MQ Operator 1.3.2 (EUS)

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.4.1

Canale operatore
`v1.3-eus`

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 solo

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.6 (canale `stable-v1`)

Novità

- Aggiunge una nuova versione di operando 9.2.0.4-r1-eus

IBM MQ Operator 1.3.1 (EUS)

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.4.1

Canale operatore
`v1.3-eus`

Valori consentiti per `.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

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 solo

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.6 (canale `stable-v1`)

Novità

- Aggiunge una nuova versione dell'operando 9.2.0.2-r1-eus

IBM MQ Operator 1.3.0 (EUS)

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.4.1

Canale operatore
`v1.3-eus`

Valori consentiti per `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2, 9.2.0.1-r1-eus

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.6 solo

IBM Cloud Pak foundational services versioni
IBM Cloud Pak foundational services 3.6 (canale`stable-v1`)

Novità

- EUS (Extended Update Support) viene offerto per campi `.spec.version` che terminano con `-eus`, quando si utilizza una licenza IBM Cloud Pak for Integration
- Aggiunge un nuovo modo di impostare le etichette e le annotazioni sulla risorsa `QueueManager` utilizzando `.spec.labels` e `.spec.annotations`

Elementi modificati

- Migliora la gestione degli errori quando si tenta di passare da una singola istanza a più istanze
- Miglioramenti alla modalità di rendering delle proprietà `QueueManager` in IBM Cloud Pak for Integration Platform Navigatore nella "Vista modulo" della console Web Red Hat OpenShift Container Platform
- Fissa la metrica di licenza predefinita quando si utilizza una licenza IBM Cloud Pak for Integration , in modo che sia `VirtualProcessorCore`
- Corregge la scheda **Risorse** per `QueueManager` nella console web di Red Hat OpenShift Container Platform , che ora mostra correttamente le risorse gestite da IBM MQ Operator per tale gestore code

Problemi e limitazioni noti

- Quando si aggiorna un `QueueManager` con un tipo di disponibilità `MultiInstance`, entrambi i pod verranno eliminati immediatamente. Entrambi dovrebbero essere riavviati rapidamente da Red Hat OpenShift Container Platform.

IBM MQ Operator 1.2.0

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.3.1

Canale operatore
`v1.2`

Valori consentiti per `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1, 9.2.0.0-r2

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.4 e successive

Novità

- Aggiunge il supporto per z /Linux
- Aggiunge condizioni di stato più dettagliate alla risorsa `QueueManager` . Per ulteriori informazioni, fare riferimento a “Condizioni di stato per QueueManager (mq.ibm.com/v1beta1)” a pagina 145
- Aggiunge ulteriori controlli di runtime per impedire l'uso di classi di memoria non valide. Per ulteriori informazioni, consultare “Disabilitazione dei controlli webhook di runtime” a pagina 114
- Semplifica l'esperienza per i gestori code a più istanze: ora è possibile scegliere con una sola proprietà (`.spec.queueManager.availability.type`) nella risorsa `QueueManager`

- Semplifica la scelta di una classe di memoria non predefinita, introducendo la proprietà `.spec.queueManager.storage.defaultClass` nella risorsa `QueueManager`

Elementi modificati

- Miglioramenti alla modalità di rendering delle proprietà `QueueManager` in IBM Cloud Pak for Integration Platform Navigatore nella "Vista modulo" della console Web Red Hat OpenShift Container Platform
- Se è disponibile una versione aggiornata del gestore code, verrà ora contrassegnata in IBM Cloud Pak for Integration Platform Navigator

IBM MQ Operator 1.1.0

CD

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.2.1

Canale operatore
`v1.1`

Valori consentiti per `.spec.version`
9.1.5.0-r2, 9.2.0.0-r1

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.4 e successive

Novità

- Aggiunge IBM MQ Advanced 9.2.0 come release di fornitura continua
- Aggiunge una funzione per specificare le informazioni INI e MQSC in una ConfigMap o Secret
- Abilita il navigatore schemi quando si utilizza la console Web Red Hat OpenShift Container Platform

Elementi modificati

- Risolve i problemi con la politica di rete, influenzando Red Hat OpenShift su IBM Cloud
- Miglioramenti all'hook web di convalida, per evitare combinazioni non valide di impostazioni nelle risorse `QueueManager`

IBM MQ Operator 1.0.0

CD

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 2020.2.1

Canale operatore
`v1.0`

Valori consentiti per `.spec.version`
9.1.5.0-r2

Red Hat OpenShift Container Platform versioni
Red Hat OpenShift Container Platform 4.4 e successive

Novità

- Versione iniziale dell'operatore, introduzione all'API `mq.ibm.com/v1beta1`

Immagini del contenitore del gestore code da utilizzare con IBM MQ Operator

9.2.5.0-r3

CD

Versione operatore richiesta
1.8.2 o superiore

Architetture supportate
amd64, s390x

Immagini

- `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`

Novità

- Novità in IBM MQ 9.2.5

Elementi modificati

- Cosa è cambiato in IBM MQ 9.2.5
- Basato su Red Hat Universal Base Image 8.6-751

9.2.5.0-r2

CD

Versione operatore richiesta
1.8.1 o superiore

Architetture supportate
amd64, s390x

Immagini

- `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`

Novità

- Novità in IBM MQ 9.2.5

Elementi modificati

- Cosa è cambiato in IBM MQ 9.2.5
- Basato su Red Hat Universal Base Image 8.5-240.1648458092

9.2.5.0-r1

CD

Versione operatore richiesta
1.8.0 o superiore

Architetture supportate
amd64, s390x

Immagini

- `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`

Novità

- Novità in IBM MQ 9.2.5

Elementi modificati

- Cosa è cambiato in IBM MQ 9.2.5
- Opzione `Gestori code remoti` non valida ora rimossa da IBM MQ Console
- Basato su Red Hat Universal Base Image 8.5-240

9.2.4.0-r1

CD

Versione operatore richiesta
1.7.0 o superiore

Architetture supportate
amd64, s390x

Immagini

- `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`

Novità

- Novità in IBM MQ 9.2.4

Elementi modificati

- Cosa è cambiato in IBM MQ 9.2.4
- Basato su Red Hat Universal Base Image 8.5-204

9.2.3.0-r1

CD

Versione operatore richiesta
1.6.0 o superiore

Architetture supportate
amd64, s390x

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.3.0-r1` (soloamd64)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.3.0-r1`
- `docker.io/ibmcom/mq:9.2.3.0-r1`

Novità

- Novità in IBM MQ 9.2.3
- Supporto per MQ Native HA per uso di produzione, quando utilizzato con una licenza IBM Cloud Pak for Integration . Si noti che i gestori code che utilizzano la HA nativa in una licenza di prova con IBM MQ 9.2.2 non possono essere aggiornati a 9.2.3. Il periodo di valutazione è terminato.

Elementi modificati

- Cosa è cambiato in IBM MQ 9.2.3
- Basato su Red Hat Universal Base Image 8.4-205

9.2.2.0-r1

CD

Versione operatore richiesta
1.5.0 o superiore

Architetture supportate
amd64, s390x

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.2.2.0-r1` (soloamd64)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.2.2.0-r1`
- `docker.io/ibmcom/mq:9.2.2.0-r1`

Novità

- Novità in IBM MQ 9.2.2
- Supporto per MQ Native HA a scopo di valutazione, quando utilizzato con una licenza IBM Cloud Pak for Integration

Elementi modificati

- Cosa è cambiato in IBM MQ 9.2.2
- Risolto un problema che causò FDC quando si arrestò un gestore code IBM MQ Advanced for Developers
- Basato su Red Hat Universal Base Image 8.3-291

9.2.1.0-r2

CD

Versione operatore richiesta
1.5.0 o superiore

Architetture supportate
amd64, s390x

Immagini

- `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`

Elementi modificati

- Corregge i problemi con SSO (single sign - on) con IBM Cloud Pak foundational services 3.7 e versioni successive.
- Basato su Red Hat Universal Base Image 8.3-291

9.2.1.0-r1

CD

Versione operatore richiesta
1.4.0 o superiore

Architetture supportate
amd64, s390x

Immagini

- `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`

Novità

- Novità in IBM MQ 9.2.1

- Le informazioni di connessione per l'instradamento predefinito sono disponibili nella console Web di MQ

Elementi modificati

- Cosa è cambiato in IBM MQ 9.2.1
- Basato su Red Hat Universal Base Image 8.3-230

9.2.0.6-r3-eus

EUS

Versione operatore richiesta
1.3.8 e futuri fix pack

Architetture supportate
amd64, s390x

Immagini

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

Elementi modificati

- Include IBM MQ 9.2.0 Fix Pack 6. Per ulteriori informazioni, consultare Fix list for IBM MQ Versione 9.2 LTS.
- Basato su Red Hat Universal Base Image 8.6-941.

9.2.0.6-r2-eus

EUS

Versione operatore richiesta
1.3.7 e futuri fix pack

Architetture supportate
amd64, s390x

Immagini

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

Elementi modificati

- Include IBM MQ 9.2.0 Fix Pack 6. Per ulteriori informazioni, consultare Fix list for IBM MQ Versione 9.2 LTS.
- Basato su Red Hat Universal Base Image 8.6-902.

9.2.0.6-r1-eus

EUS

Versione operatore richiesta
1.3.6 e futuri fix pack

Architetture supportate
amd64, s390x

Immagini

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

Elementi modificati

- Include IBM MQ 9.2.0 Fix Pack 6. Per ulteriori informazioni, consultare Fix list for IBM MQ Versione 9.2 LTS.
- Basato su Red Hat Universal Base Image 8.6-854.

9.2.0.5-r3-eus

EUS

Versione operatore richiesta
1.3.5 e fix pack futuri

Architetture supportate
amd64, s390x

Immagini

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

Elementi modificati

- Include IBM MQ 9.2.0 Fix Pack 5. Per ulteriori informazioni, consultare What's changed in IBM MQ 9.2.0 Fix Pack 5 e l'elenco di correzioni per IBM MQ Versione 9.2 LTS.
- Basato su Red Hat Universal Base Image 8.6-751.1655117800.

9.2.0.5-r2-eus

EUS

Versione operatore richiesta
1.3.4 e futuri fix pack

Architetture supportate
amd64, s390x

Immagini

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

Elementi modificati

- Include IBM MQ 9.2.0 Fix Pack 5. Per ulteriori informazioni, vedi What's changed in IBM MQ 9.2.0 Fix Pack 5 e Fix list for IBM MQ Versione 9.2 LTS
- Basato su Red Hat Universal Base Image 8.6-751

9.2.0.5-r1-eus

EUS

Versione operatore richiesta
1.3.3 e futuri fix pack

Architetture supportate
amd64, s390x

Immagini

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

Elementi modificati

- Include IBM MQ 9.2.0 Fix Pack 5. Per ulteriori informazioni, vedi What's changed in IBM MQ 9.2.0 Fix Pack 5 e Fix list for IBM MQ Versione 9.2 LTS
- Basato su Red Hat Universal Base Image 8.5-240.1648458092

9.2.0.4-r1-eus

EUS

Versione operatore richiesta
1.3.2 e futuri fix pack

Architetture supportate
amd64, s390x

Immagini

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

Elementi modificati

- Include IBM MQ 9.2.0 Fix Pack 4. Per ulteriori informazioni, vedi What's changed in IBM MQ 9.2.0 Fix Pack 4 e Fix list for IBM MQ Versione 9.2 LTS
- Basato su Red Hat Universal Base Image 8.5-204

9.2.0.2-r2-eus

Versione operatore richiesta
1.6.0 o superiore

Architetture supportate
amd64, s390x

Immagini

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

Elementi modificati

- Risolve i problemi con SSO (single sign - on) con IBM Cloud Pak foundational services 3.7 e versioni successive, necessari solo quando si esegue la migrazione da una release EUS a una release CD.
- Basato su Red Hat Universal Base Image 8.4-200.1622548483

9.2.0.2-r1-eus

Versione operatore richiesta
1.3.1 e futuri fix pack ; 1.6.0 o superiore

Architetture supportate
amd64, s390x

Immagini

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

Elementi modificati

- L'integrazione di Operations Dashboard utilizza l'agent di traccia e il raccoglitore versione 1.0.8
- Include IBM MQ 9.2.0 Fix Pack 2. Per ulteriori informazioni, vedi What's changed in IBM MQ 9.2.0 Fix Pack 2 e Fix list for IBM MQ Versione 9.2 LTS
- Basato su Red Hat Universal Base Image 8.4-200.1622548483

9.2.0.1-r1-eus

Versione operatore richiesta
1.3.0 o superiore

Architetture supportate
amd64, s390x

Immagini

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

Novità

- Disponibile solo quando si utilizza una licenza IBM Cloud Pak for Integration

- ESS (Extended Update Support) è disponibile quando si utilizza IBM MQ Operator 1.3.x e IBM Common Services 3.6, su Red Hat OpenShift Container Platform 4.6

Elementi modificati

- Include IBM MQ 9.2.0 Fix Pack 1. Per ulteriori informazioni, vedi What's changed in IBM MQ 9.2.0 Fix Pack 1 e Fix list for IBM MQ Versione 9.2 LTS
- Basato su Red Hat Universal Base Image 8.3-201
- Corregge i problemi con il probe di attività (`chkmqhealthy`) e il probe di disponibilità (`chkmqready`) durante l'esecuzione in `SecurityContextContraints` che consentono l'escalation dei privilegi.

9.2.0.0-r3

CD

Versione operatore richiesta
1.5.0 o superiore

Architetture supportate
amd64, s390x

Immagini

- `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`

Elementi modificati

- Basato su Red Hat Universal Base Image 8.3-291

9.2.0.0-r2

CD

Versione operatore richiesta
1.2.0 o superiore

Architetture supportate
amd64, s390x

Immagini

- `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`

Novità

- Ora disponibile su z /Linux

Elementi modificati

- Basato su Red Hat Universal Base Image 8.2-349

9.2.0.0-r1

CD

Versione operatore richiesta
1.1.0 o superiore

Architetture supportate
amd64

Immagini

- `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`

Novità

- Novità in IBM MQ 9.2.0

Elementi modificati

- Cosa è cambiato in IBM MQ 9.2.0
- Utilizza l'argomento `-ic` per `crtmqm` per applicare automaticamente i file MQSC. Sostituisce l'utilizzo precedente dei comandi `runmqsc`
- Basato su Red Hat Universal Base Image 8.2-301.1593113563

9.1.5.0-r2

Versione operatore richiesta
1.0.0 o superiore

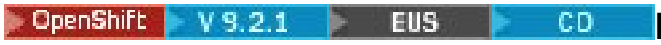
Architetture supportate
amd64

Immagini

- `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`

Elementi modificati

- Basato su Red Hat Universal Base Image 8.2-267

 Migrazione di IBM MQ a IBM Cloud Pak for Integration

Questa serie di argomenti descrive i passi chiave per la migrazione di un gestore code IBM MQ esistente in un ambiente contenitore utilizzando IBM MQ Operator in IBM Cloud Pak for Integration.

Informazioni su questa attività

I client che distribuiscono IBM MQ su Red Hat OpenShift possono essere separati nei seguenti scenari:

1. Creazione di una nuova distribuzione IBM MQ in Red Hat OpenShift per nuove applicazioni.
2. Estensione di una rete IBM MQ in Red Hat OpenShift per nuove applicazioni in Red Hat OpenShift.
3. Spostamento di una distribuzione IBM MQ in Red Hat OpenShift per continuare a supportare le applicazioni esistenti.

È solo per lo scenario 3 che è necessario eseguire la migrazione della configurazione IBM MQ . Gli altri scenari sono considerati nuove distribuzioni.

Questa serie di argomenti è incentrata sullo scenario 3 e descrive i passi chiave per migrare un gestore code IBM MQ esistente in un ambiente contenitore utilizzando IBM MQ Operator. A causa della flessibilità e dell'ampio uso di IBM MQ, ci sono diversi passi facoltativi. Ognuno di questi include una sezione "Ho bisogno di fare questo". La verifica delle proprie esigenze consente di risparmiare tempo durante la migrazione.

È inoltre necessario considerare quali dati migrare:

1. Migrare IBM MQ con la stessa configurazione ma senza alcun messaggio accodato esistente.
2. Migrare IBM MQ con la stessa configurazione e messaggi esistenti.

Una tipica migrazione da versione a versione può utilizzare entrambi gli approcci. In un tipico gestore code IBM MQ nel punto di migrazione, vi sono pochi messaggi memorizzati nelle code, il che rende l'opzione 1 appropriata per molti casi. In caso di migrazione a una piattaforma contenitore, è ancora più comune utilizzare l'opzione 1, per ridurre la complessità della migrazione e consentire una distribuzione verde blu. Pertanto, le istruzioni si concentrano su questo scenario.

L'obiettivo di questo scenario è creare un gestore code nell'ambiente del contenitore che corrisponda alla definizione del gestore code esistente. Ciò consente alle applicazioni collegate alla rete esistenti di essere semplicemente riconfigurate per puntare al nuovo gestore code, senza modificare alcuna altra configurazione o logica dell'applicazione.

Durante questa migrazione si generano più file di configurazione da applicare al nuovo gestore code. Per semplificare la gestione di questi file, è necessario creare una directory e generarli in tale directory.

Procedura

1. “Verifica della disponibilità delle funzioni richieste” a pagina 37
2. “Estrazione della configurazione del gestore code” a pagina 38
3. Opzionale: “Facoltativo: estrazione e acquisizione delle chiavi e dei certificati del gestore code” a pagina 38
4. Opzionale: “Facoltativo: configurazione di LDAP” a pagina 40
5. Opzionale: “Facoltativo: modifica degli indirizzi IP e dei nomi host nella configurazione IBM MQ” a pagina 47
6. “Aggiornamento della configurazione del gestore code per un ambiente contenitore” a pagina 49
7. “Selezione dell'architettura HA di destinazione per IBM MQ in esecuzione nei contenitori” a pagina 51
8. “Creazione delle risorse per il gestore code” a pagina 52
9. “Creazione del nuovo gestore code su Red Hat OpenShift” a pagina 53
10. “Verifica della nuova distribuzione del contenitore” a pagina 57

OpenShift V 9.2.1 EUS CD Verifica della disponibilità delle funzioni richieste

IBM MQ Operator non include tutte le funzioni disponibili in IBM MQ Advanceded è necessario verificare che tali funzioni non siano richieste. Altre funzioni sono parzialmente supportate e possono essere riconfigurate in modo da corrispondere a quanto disponibile nel contenitore.

Prima di iniziare

Questo è il primo passo in “Migrazione di IBM MQ a IBM Cloud Pak for Integration” a pagina 36.

Procedura

1. Verificare che l'immagine del contenitore di destinazione includa tutte le funzioni richieste.

   Per le informazioni più recenti, consultare “Scelta della modalità di utilizzo di IBM MQ nei contenitori” a pagina 5.
2. IBM MQ Operator ha una singola porta di traffico IBM MQ , nota come listener. Se si dispone di più listener, semplificarlo per utilizzare un singolo listener nel contenitore. Poiché questo non è uno scenario comune, questa modifica non viene documentata in dettaglio.
3. Se vengono utilizzate le uscite IBM MQ , eseguirne la migrazione nel contenitore eseguendo il layering nei file binari di uscita IBM MQ . Questo è uno scenario di migrazione avanzata e quindi non incluso qui. Per una descrizione della procedura, vedere “Creazione di un'immagine con file MQSC e INI personalizzati, utilizzando la CLI Red Hat OpenShift” a pagina 112.

4. Se il sistema IBM MQ include l'alta disponibilità, esaminare le opzioni disponibili.

   Vedere “Alta disponibilità per IBM MQ nei contenitori” a pagina 16.

Operazioni successive

È ora possibile estrarre la configurazione del gestore code.

OpenShift V 9.2.1 EUS CD Estrazione della configurazione del gestore code

La maggior parte della configurazione è portabile tra gestori code. Ad esempio, gli elementi con cui interagiscono le applicazioni, come le definizioni di code, argomenti e canali. Utilizzare questa attività per estrarre la configurazione dal gestore code IBM MQ esistente.

Prima di iniziare

Questa attività presuppone che sia stato verificato che le funzioni richieste siano disponibili.

Procedura

1. Accedere alla macchina con l'installazione esistente di IBM MQ .
2. Eseguire il backup della configurazione.

   Esegui il seguente comando:

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

   Note di utilizzo per questo comando:

   - Questo comando memorizzi il backup nella directory `tmp` . È possibile memorizzare il backup in un'altra posizione, ma questo scenario presuppone la directory `tmp` per i comandi successivi.
   - Sostituisci *QMGR_NAME* con il nome del gestore code dal tuo ambiente. Se non si è certi del valore, eseguire il comando **dspmq** per visualizzare i gestori code disponibili sulla macchina. Di seguito è riportato l'output del comando **dspmq** di esempio per il gestore code denominato `qm1`:

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

     Il comando **dspmq** richiede l'avvio del gestore code IBM MQ , altrimenti si riceve il seguente errore:

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

     Se necessario, avviare il gestore code immettendo il seguente comando:

     ```
     strmqm QMGR_NAME
     ```

Operazioni successive

Sei ora pronto a estrarre e acquisire le chiavi e i certificati del gestore code.

OpenShift V 9.2.1 EUS CD Facoltativo: estrazione e acquisizione delle chiavi e dei certificati del gestore code

IBM MQ può essere configurato utilizzando TLS per crittografare il traffico nel gestore code. Utilizzare questa attività per verificare se il gestore code sta utilizzando TLS, per estrarre chiavi e certificati e per configurare TLS sul gestore code migrato.

Prima di iniziare

Questa attività presuppone che sia stata estratta la configurazione del gestore code.

Informazioni su questa attività

È necessario?

IBM MQ può essere configurato per crittografare il traffico nel gestore code. Questa codifica viene completata utilizzando un repository delle chiavi configurato nel gestore code. I canali IBM MQ quindi abilitano la comunicazione TLS. Se non si è sicuri che sia configurato nel proprio ambiente, eseguire il seguente comando per verificare:

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

Se non viene trovato alcun risultato, TLS non viene utilizzato. Tuttavia, ciò non significa che TLS non debba essere configurato nel gestore code migrato. Esistono diversi motivi per cui si potrebbe voler modificare questo comportamento:

- L'approccio di sicurezza nell'ambiente Red Hat OpenShift deve essere migliorato rispetto all'ambiente precedente.
- Se devi accedere al gestore code migrato dall'esterno dell'ambiente Red Hat OpenShift , TLS è richiesto per passare attraverso l'instradamento Red Hat OpenShift .

Procedura

1. Estrarre i certificati attendibili dall'archivio esistente.

   Se TLS è attualmente in uso sul gestore code, il gestore code potrebbe avere un numero di certificati attendibili memorizzati. Questi devono essere estratti e copiati nel nuovo gestore code. Completare una delle seguenti operazioni facoltative:

   - Per semplificare l'estrazione dei certificati, eseguire il seguente script sul sistema locale:

     ```
     #!/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
     ```

     Quando si esegue lo script, specificare l'ubicazione del backup IBM MQ come argomento e i certificati vengono estratti. Ad esempio, se lo script è denominato `extractCert.sh` e il backup di IBM MQ si trova in `/tmp/backup.mqsc` , eseguire il seguente comando:

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

   - In alternativa, eseguire i seguenti comandi nell'ordine mostrato:

     a. Identificare l'ubicazione dell'archivio TLS:

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

     Output di esempio:

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

dove il keystore si trova in `/run/runmqserver/tls/key.kdb`

b. In base a queste informazioni sull'ubicazione, interrogare il keystore per determinare i certificati archiviati:

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

Output di esempio:

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

c. Estrarre ciascuno dei certificati elencati. Eseguire questa operazione immettendo il seguente comando:

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

Nei campioni precedentemente mostrati, ciò equivale a quanto segue:

```
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. Acquisire una nuova chiave e un certificato per il gestore code

Per configurare TLS sul gestore code migrato, generare una nuova chiave e un nuovo certificato. Viene quindi utilizzato durante la distribuzione. In molte organizzazioni ciò significa contattare il team di sicurezza per richiedere una chiave e un certificato. In alcune organizzazioni questa opzione non è disponibile e vengono utilizzati i certificati autofirmati.

Il seguente esempio genera un certificato autofirmato in cui la scadenza è impostata su 10 anni:

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

Vengono creati due nuovi file:

- `qmgr.key` è la chiave privata per il gestore code
- `qmgr.crt` è il certificato pubblico

Operazioni successive

Ora è possibile configurare LDAP.

OpenShift V 9.2.1 EUS CD Facoltativo: configurazione di LDAP

IBM MQ Operator può essere configurato per utilizzare diversi approcci di sicurezza. Di solito LDAP è il più efficace per una distribuzione enterprise e LDAP viene utilizzato per questo scenario di migrazione.

Prima di iniziare

Questa attività presuppone che l'utente abbia estratto e acquisito le chiavi e i certificati del gestore code.

Informazioni su questa attività

È necessario?

Se si sta già utilizzando LDAP per l'autenticazione e l'autorizzazione, non è richiesta alcuna modifica.

Se non si è certi dell'utilizzo di LDAP, eseguire il seguente comando:

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

Output di esempio:

```
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
```

Nell'output sono presenti due attributi di particolare interesse:

AUTHTYPE
Se ha il valore IDPWLDAP, si sta utilizzando LDAP per l'autenticazione.

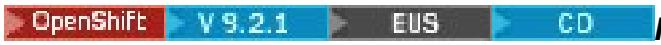
Se il valore è vuoto o un altro valore, LDAP non è configurato. In questo caso, controllare l'attributo AUTHORMD per verificare se gli utenti LDAP vengono utilizzati per l'autorizzazione.

AUTHORMD
Se questo ha il valore OS, non si sta utilizzando LDAP per l'autorizzazione.

Per modificare l'autenticazione e l'autorizzazione per utilizzare LDAP, completare le seguenti attività:

Procedura

1. Aggiornare il backup IBM MQ per il server LDAP.
2. Aggiornare il backup IBM MQ per informazioni di autorizzazione LDAP.



Parte 1 LDAP: aggiornamento del backup IBM MQ per il server LDAP

Una descrizione completa di come impostare LDAP non rientra nell'ambito di questo scenario. Questo argomento fornisce un riepilogo del processo, un esempio e riferimenti a ulteriori informazioni.

Prima di iniziare

Questa attività presuppone che l'utente abbia estratto e acquisito le chiavi e i certificati del gestore code.

Informazioni su questa attività

È necessario?

Se si sta già utilizzando LDAP per l'autenticazione e l'autorizzazione, non è richiesta alcuna modifica. Se non si è certi dell'utilizzo di LDAP, consultare “Facoltativo: configurazione di LDAP” a pagina 40.

Ci sono due parti per impostare il server LDAP:

1. Definire una configurazione LDAP.
2. Associare la configurazione LDAP alla definizione del gestore code.

Ulteriori informazioni di supporto per questa configurazione:

- Panoramica repository utente
- Guida di riferimento al comando AUTHINFO

Procedura

1. Definire una configurazione LDAP.

   Modificare il file di `backup.mqsc` per definire un nuovo oggetto **AUTHINFO** per il sistema LDAP. Ad esempio:

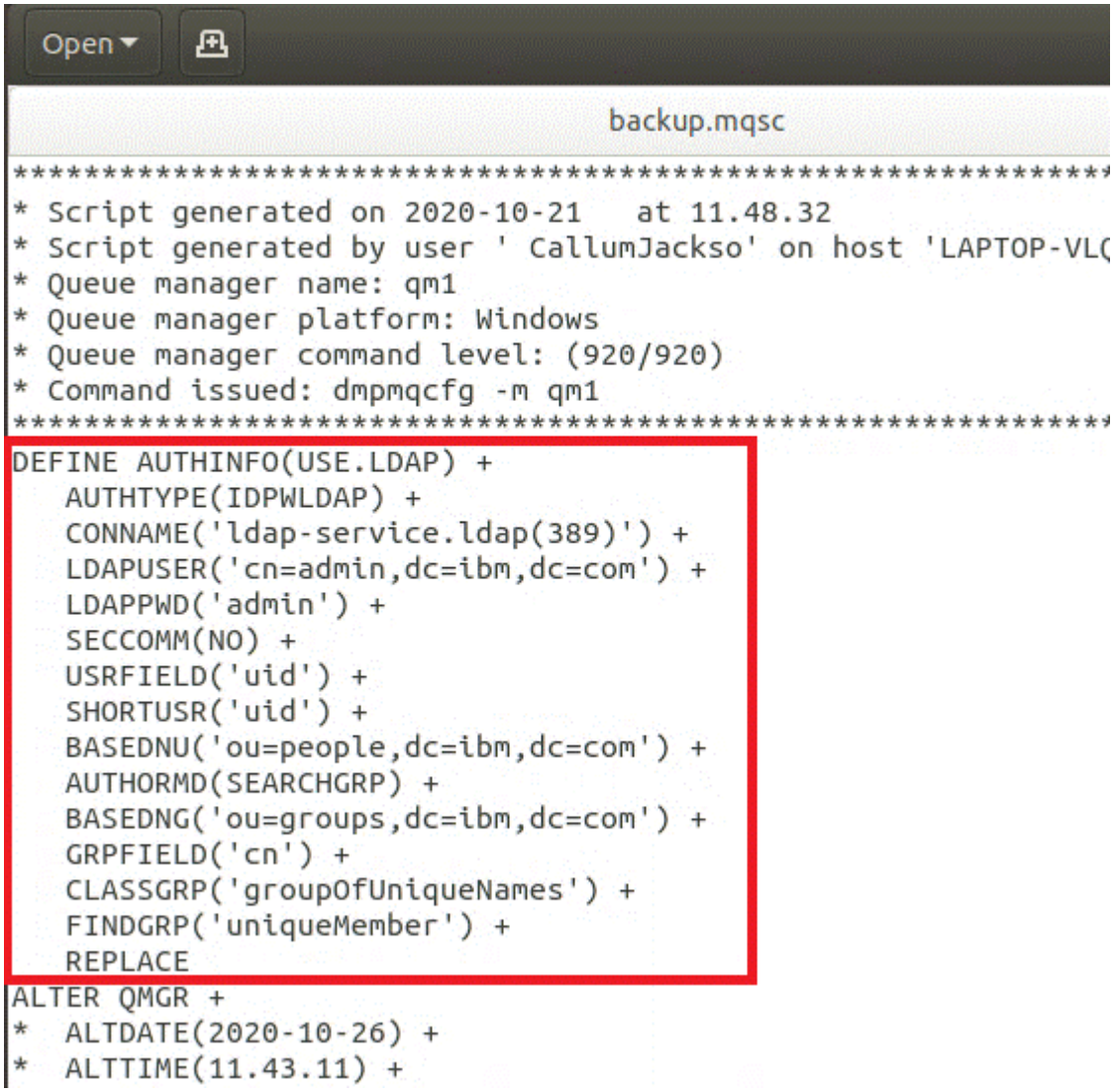
   ```
   DEFINE AUTHINFO(USE.LDAP) +
      AUTHTYPE(IDPWLDAP) +
      CONNAME('ldap-service.ldap(389)') +
      LDAPUSER('cn=admin,dc=ibm,dc=com') +
      LDAPPWD('admin') +
      SECCOMM(NO) +
      USRFIELD('uid') +
      SHORTUSR('uid') +
      BASEDNU('ou=people,dc=ibm,dc=com') +
      AUTHORMD(SEARCHGRP)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

   dove

   - **CONNAME** è il nome host e la porta corrispondenti al server LDAP. Se esistono più indirizzi per la resilienza, questi possono essere configurati utilizzando un elenco separato da virgole.
   - **LDAPUSER** è il DN (distinguished name) corrispondente all'utente che IBM MQ utilizza durante la connessione a LDAP per interrogare i record utente.
   - **LDAPPWD** è la password che corrisponde all'utente **LDAPUSER** .
   - **SECCOM** specifica se la comunicazione con il server LDAP deve utilizzare TLS. I valori possibili sono:
     - YES: viene utilizzato TLS e il server IBM MQ presenta un certificato.
     - ANON: TLS viene utilizzato senza un certificato presentato dal server IBM MQ .
     - NO: TLS non viene utilizzato durante la connessione.
   - **USRFIELD** specifica il campo nel record LDAP rispetto al quale deve corrispondere il nome utente presentato.
   - **SHORTUSR** è un campo all'interno del record LDAP che non supera i 12 caratteri di lunghezza. Il valore all'interno di questo campo è l'identità asserita se l'autenticazione ha esito positivo.
   - **BASEDNU** è il DN di base da utilizzare per la ricerca LDAP.
   - **BASEDNG** è il DN di base per i gruppi all'interno di LDAP.
   - **AUTHORMD** definisce il meccanismo utilizzato per risolvere l'appartenenza al gruppo per l'utente. Esistono quattro opzioni:
     - SO: interrogare il sistema operativo per i gruppi associati al nome breve.
     - SEARCHGRP: ricercare le voci del gruppo in LDAP per l'utente autenticato.
     - SEARCHUS: ricercare nel record utente autenticato le informazioni di appartenenza al gruppo.
     - SRCHGRPSN: ricercare le voci gruppo in LDAP per il nome utente breve degli utenti autenticati (definito dal campo SHORTUSR).
   - **GRPFIELD** è l'attributo all'interno del record del gruppo LDAP che corrisponde a un nome semplice. Se specificato, può essere utilizzato per definire i record di autorizzazione.

- **CLASSUSR** è la classe oggetto LDAP che corrisponde a un utente.
- **CLASSGRP** è la classe di oggetti LDAP che corrisponde a un gruppo.
- **FINDGRP** è l'attributo all'interno del record LDAP che corrisponde all'appartenenza al gruppo.

La nuova voce può essere posizionata in qualsiasi punto all'interno del file, tuttavia potrebbe essere utile avere delle nuove voci all'inizio del file:



2. Associare la configurazione LDAP alla definizione del gestore code.

   È necessario associare la configurazione LDAP alla definizione del gestore code. Immediatamente sotto la voce `DEFINE AUTHINFO` è presente una voce `ALTER QMGR`. Modificare la voce CONNAUTH in modo che corrisponda al nome AUTHINFO appena creato. Ad esempio, nell'esempio precedente, `AUTHINFO(USE.LDAP)` è stato definito, il che significa che il nome è `USE.LDAP`. Modificare quindi `CONNAUTH('SYSTEM.DEFAULT.AUTHINFO.IDPWOS')` in `CONNAUTH('USE.LDAP')`:

Per fare in modo che il passaggio a LDAP avvenga immediatamente, richiamare un comando REFRESH SECURITY aggiungendo una riga immediatamente dopo il comando `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
```

Operazioni successive

Ora è possibile aggiornare il backup IBM MQ per le informazioni di autorizzazione LDAP.

OpenShift V 9.2.1 EUS CD

Parte LDAP 2: aggiornamento del backup IBM MQ per le informazioni di autorizzazione LDAP

IBM MQ fornisce regole di autorizzazione dettagliate che controllano l'accesso agli oggetti IBM MQ . Se l'autenticazione e l'autorizzazione sono state modificate in LDAP, le regole di autorizzazione potrebbero non essere valide e richiedere l'aggiornamento.

Prima di iniziare

Questa attività presuppone che sia stato aggiornato il backup per il server LDAP.

Informazioni su questa attività

È necessario?

Se si sta già utilizzando LDAP per l'autenticazione e l'autorizzazione, non è richiesta alcuna modifica. Se non si è certi dell'utilizzo di LDAP, consultare “Facoltativo: configurazione di LDAP” a pagina 40.

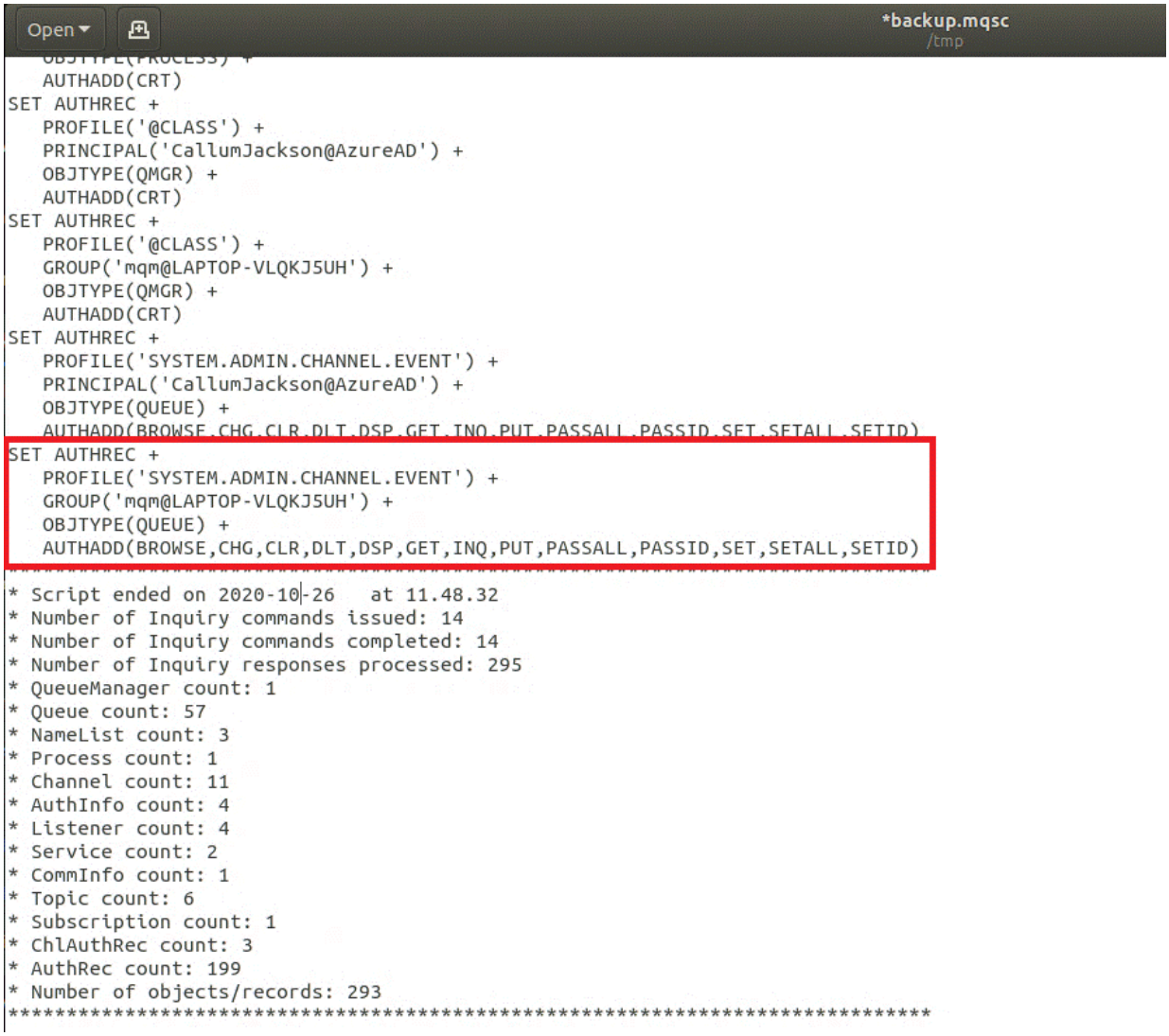
Esistono due parti per aggiornare le informazioni di autorizzazione LDAP:

1. Rimuovere tutte le autorizzazioni esistenti dal file.
2. Definire nuove informazioni di autorizzazione per LDAP.

Procedura

1. Rimuovere tutte le autorizzazioni esistenti dal file.

   Nel file di backup, vicino alla fine del file, vengono visualizzate diverse voci che iniziano con SET AUTHREC:



   Individuare le voci esistenti ed eliminarle. L'approccio più semplice consiste nel rimuovere tutte le regole SET AUTHREC esistenti, quindi creare nuove voci basate sulle voci LDAP.
2. Definire nuove informazioni di autorizzazione per LDAP

A seconda della configurazione del gestore code e del numero di risorse e gruppi, questa potrebbe essere un'attività che richiede tempo o semplice. Nel seguente esempio si assume che il gestore code abbia solo una singola coda denominata Q1e si desidera consentire l'accesso al gruppo LDAP apps .

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

Il primo comando AUTHREC aggiunge l'autorizzazione per accedere al gestore code e il secondo fornisce l'accesso alla coda. Se è richiesto l'accesso a una seconda coda, è necessario un terzo comando AUTHREC , a meno che non si decida di utilizzare i caratteri jolly per fornire un accesso più generico.

Ecco un altro esempio. Se un gruppo di amministratori (denominato `admins`) ha bisogno di accesso completo al gestore code, aggiungere i seguenti comandi:

```
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)
```

Operazioni successive

Sei ora pronto a modificare gli indirizzi IP e i nomi host nella IBM MQ configurazione.

OpenShift V 9.2.1 EUS CD Facoltativo: modifica degli indirizzi IP e dei nomi host nella configurazione IBM MQ

Per la configurazione di IBM MQ potrebbero essere specificati indirizzi IP e nomi host. In alcune situazioni queste possono rimanere, mentre in altre situazioni devono essere aggiornate.

Prima di iniziare

Questa attività presuppone che sia stato configurato LDAP.

Informazioni su questa attività

È necessario?

Innanzitutto, determinare se si dispone di indirizzi IP o nomi host specificati, a parte la configurazione LDAP definita nella sezione precedente. A tale scopo, eseguire il seguente comando:

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

Output di esempio:

```
*****************************************************
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(' ') +
```

In questo esempio la ricerca restituisce tre risultati. Un risultato corrisponde alla configurazione LDAP definita in precedenza. Questo può essere ignorato, perché il nome host del server LDAP rimane lo stesso. Gli altri due risultati sono voci di connessione vuote, quindi possono essere ignorati. Se non si dispone di voci aggiuntive, è possibile ignorare il resto di questo argomento.

Procedura

1. Comprendere le voci restituite.

   IBM MQ può includere indirizzi IP, nomi host e porte in molti aspetti della configurazione. Possiamo classificarli in due categorie:

   a. **Posizione di questo gestore code**: le informazioni sull'ubicazione che questo gestore code utilizza o pubblica, che altri gestori code o applicazioni all'interno di una rete IBM MQ possono utilizzare per la connettività.

   b. **Ubicazione delle dipendenze del gestore code**: le ubicazioni di altri gestori code o sistemi di cui questo gestore code deve essere a conoscenza.

   Poiché questo scenario è concentrato solo sulle modifiche a questa configurazione del gestore code, gestiamo solo gli aggiornamenti di configurazione per la categoria (a). Tuttavia, se a questa ubicazione del gestore code fanno riferimento altri gestori code o applicazioni, potrebbe essere necessario aggiornare le relative configurazioni per corrispondere alla nuova ubicazione di questo gestore code.

   Ci sono due oggetti chiave che potrebbero contenere informazioni che devono essere aggiornate:

   - Listener: rappresentano l'indirizzo di rete su cui è in ascolto IBM MQ .
   - CLUSTER RECEIVER canale: se il gestore code fa parte di un cluster IBM MQ , questo oggetto esiste. Specifica l'indirizzo di rete a cui altri gestori code possono connettersi.

2. Nell'output originale del comando `grep 'CONNAME\|LOCLADDR\|IPADDRV' -B 3 backup.mqsc` , identificare se sono definiti canali CLUSTER RECEIVER. In caso affermativo, aggiornare gli indirizzi IP.

   Per identificare se sono definiti canali CLUSTER RECEIVER, individuare le voci con CHLTYPE (CLUSRCVR) nell'output originale:

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

   Se le voci esistono, aggiornare CONNAME con l'instradamento IBM MQ Red Hat OpenShift . Questo valore è basato sull'ambiente Red Hat OpenShift e utilizza una sintassi prevedibile:

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

   Ad esempio, se la distribuzione del gestore code è denominata qm1 all'interno dello spazio dei nomi cp4i e *openshift_app_route_hostname* è `apps.callumj.icp4i.com`, l'URL di instradamento è il seguente:

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

   Il numero di porta per l'instradamento è generalmente 443. A meno che l'amministratore di Red Hat OpenShift non lo dica in modo diverso, questo è di solito il valore corretto. Utilizzando queste informazioni, aggiornare i campi CONNAME . Ad esempio:

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

Nell'output originale del comando `grep 'CONNAME\|LOCLADDR\|IPADDRV' -B 3 backup.mqsc`, verificare se esistono voci per `LOCLADDR` o `IPADDRV`. Se lo fanno, eliminarli. Non sono rilevanti in un ambiente contenitore.

Operazioni successive

Si è ora pronti ad aggiornare la configurazione del gestore code per un ambiente contenitore.

Aggiornamento della configurazione del gestore code per un ambiente contenitore

Quando è in esecuzione in un contenitore, alcuni aspetti della configurazione sono definiti dal contenitore e potrebbero essere in conflitto con la configurazione esportata.

Prima di iniziare

Questa attività presuppone che si disponga di ha modificato la configurazione IBM MQ di indirizzi IP e nomi host.

Informazioni su questa attività

I seguenti aspetti di configurazione sono definiti dal contenitore:

- Le definizioni del listener (che corrispondono alle porte esposte).
- L'ubicazione di qualsiasi archivio TLS potenziale.

Pertanto, è necessario aggiornare la configurazione esportata:

1. Rimuovere tutte le definizioni di listener.
2. Definire l'ubicazione del repository delle chiavi TLS.

Procedura

1. Rimuovere tutte le definizioni di listener.

   Nella configurazione di backup, cercare `DEFINE LISTENER`. Deve essere compreso tra le definizioni `AUTHINFO` e `SERVICE` . Evidenziare l'area ed eliminarla.

*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. Definire l'ubicazione del repository chiavi TLS.

   Il backup del gestore code contiene la configurazione TLS per l'ambiente originale. Questo è diverso dall'ambiente del contenitore e quindi sono necessari un paio di aggiornamenti:

   - Modificare la voce **CERTLABL** in `default`
   - Modificare l'ubicazione del repository delle chiavi TLS (**SSLKEYR**) in `/run/runmqserver/tls/key`

   Per trovare l'ubicazione dell'attributo **SSLKEYR** nel file, ricercare **SSLKEYR**. Di solito viene trovata una sola voce. Se vengono trovate più voci, verificare che si stia modificando l'oggetto **QMGR** come mostrato nella seguente figura:

*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
```

Operazioni successive

Sei ora pronto a selezionare l'architettura di destinazione per IBM MQ in esecuzione nei contenitori.

OpenShift V 9.2.1 EUS CD

Selezione dell'architettura HA di destinazione per IBM MQ in esecuzione nei contenitori

Scegli tra una singola istanza (un singolo pod Kubernetes) e una multi - istanza (due pod) per soddisfare i tuoi requisiti di alta disponibilità.

Prima di iniziare

Questa attività presuppone che sia stata aggiornata la configurazione del gestore code per un ambiente contenitore.

Informazioni su questa attività

IBM MQ Operator fornisce due opzioni di alta disponibilità:

- **Istanza singola**: viene avviato un singolo contenitore (Pod) ed è responsabilità di Red Hat OpenShift riavviare in caso di errore. A causa delle caratteristiche di una serie con stato all'interno di Kubernetes, ci sono diverse situazioni in cui questo failover potrebbe richiedere un periodo di tempo prolungato o richiedere il completamento di un'azione amministrativa.
- **A più istanze**: vengono avviati due contenitori (ciascuno in un pod separato), uno in modalità attiva e un altro in standby. Questa topologia consente un failover molto più rapido. Richiede un file system Read Write Many che soddisfi i requisiti IBM MQ .

In questa attività si sceglie solo l'architettura HA di destinazione. I passi per la configurazione dell'architettura scelta sono descritti in un'attività successiva in questo scenario (“Creazione del nuovo gestore code su Red Hat OpenShift” a pagina 53).

Procedura

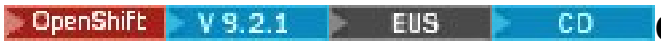
1. Esaminare le due opzioni.

   Per una descrizione completa di queste opzioni, consultare “Alta disponibilità per IBM MQ nei contenitori” a pagina 16.
2. Selezionare l'architettura HA di destinazione.

   Se non sei sicuro di quale opzione scegliere, inizia con l'opzione **Singola istanza** e verifica se soddisfa i tuoi requisiti di alta disponibilità.

Operazioni successive

Si è ora pronti a creare le risorse del gestore code.



Creazione delle risorse per il gestore code

Importa la configurazione di IBM MQ e i certificati e chiavi TLS nell'ambiente Red Hat OpenShift .

Prima di iniziare

Questa attività presuppone che si disponga di l'architettura di destinazione selezionata per IBM MQ in esecuzione nei contenitori.

Informazioni su questa attività

Nelle sezioni precedenti sono state estratte, aggiornate e definite due risorse:

- IBM MQ configurazione
- Chiavi e certificati TLS

È necessario importare queste risorse nell'ambiente Red Hat OpenShift prima che il gestore code venga distribuito.

Procedura

1. Importare la configurazione IBM MQ in Red Hat OpenShift.

Le seguenti istruzioni presuppongono che si disponga della configurazione IBM MQ nella directory corrente, in un file denominato `backup.mqsc`. Altrimenti, è necessario personalizzare il nome file in base al proprio ambiente.

a) Accedere al cluster utilizzando `oc login`.

b) Caricare la configurazione IBM MQ in un `configmap`.

Esegui il seguente comando:

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

c) Verificare che il file sia stato caricato correttamente.

Esegui il seguente comando:

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

2. Importare le IBM MQ risorse TLS

Come discusso in "Facoltativo: estrazione e acquisizione delle chiavi e dei certificati del gestore code" a pagina 38, TLS potrebbe essere richiesto per la distribuzione del gestore code. In tal caso, è necessario disporre già di un numero di file che terminano con `.crt` e `.key`. È necessario aggiungerli ai segreti Kubernetes per il gestore code a cui fare riferimento al momento della distribuzione.

Ad esempio, se si dispone di una chiave e di un certificato per il gestore code, potrebbero essere richiamati:

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

Per importare questi file, eseguire il seguente comando:

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

Kubernetes fornisce questo utile programma di utilità quando stai importando una chiave pubblica e privata corrispondente. Se si dispone di ulteriori certificati da aggiungere, ad esempio nel truststore del gestore code, eseguire il seguente comando:

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

Ad esempio, se i file da importare sono `trust1.crt`, `trust2.crt` e `trust3.crt`, il comando è il seguente:

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

Operazioni successive

Ora è possibile creare il nuovo gestore code su Red Hat OpenShift.

OpenShift V 9.2.1 EUS CD Creazione del nuovo gestore code su Red Hat OpenShift

Distribuire una singola istanza o un gestore code a più istanze su Red Hat OpenShift.

Prima di iniziare

Questa attività presuppone che l'utente abbia creato le risorse del gestore codee abbia installato IBM MQ Operator in Red Hat OpenShift.

Informazioni su questa attività

Come descritto in “Selezione dell'architettura HA di destinazione per IBM MQ in esecuzione nei contenitori” a pagina 51, esistono due possibili topologie di distribuzione. Pertanto, questo argomento fornisce due diversi modelli:

- Distribuire un gestore code a istanza singola.
- Distribuire un gestore code a più istanze.

Importante: Completare solo uno dei due modelli, in base alla topologia preferita.

Procedura

- Distribuire un gestore code a istanza singola.

  Il gestore code migrato viene distribuito a Red Hat OpenShift utilizzando un file YAML. Di seguito è riportato un esempio, basato sui nomi utilizzati negli argomenti precedenti:

```
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
```

  A seconda dei passaggi che hai eseguito, potrebbe essere necessario personalizzare il precedente YAML. Per aiutarvi con questo, ecco una spiegazione di questo YAML:

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

  Definisce l'oggetto, il tipo e il nome Kubernetes . L'unico campo che richiede la personalizzazione è quello `name` .

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

Ciò corrisponde alle informazioni sulla versione e sulla licenza per la distribuzione. Se è necessario personalizzarlo, utilizzare le informazioni fornite in "Riferimento per la licenza per mq.ibm.com/v1beta1" a pagina 126.

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

Perché il gestore code sia configurato per utilizzare TLS, deve fare riferimento ai certificati e alle chiavi pertinenti. Il campo `secretName` fa riferimento al segreto Kubernetes creato nella sezione Importa le risorse IBM MQ TLS e l'elenco di elementi (`tls.key` e `tls.crt`) sono i nomi standard assegnati da Kubernetes quando si utilizza la sintassi `oc create secret tls` . Se si dispone di ulteriori certificati da aggiungere al truststore, è possibile aggiungerli in modo simile, ma gli elementi sono i nomi file corrispondenti utilizzati durante l'importazione. Ad esempio, il seguente codice può essere utilizzato per creare i certificati del truststore:

```
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
```

Importante: Se TLS non è obbligatorio, eliminare la sezione TLS di YAML.

```
  web:
    enabled: true
```

Ciò abilita la console Web per la distribuzione

```
  queueManager:
    name: QM1
```

Definisce il nome del gestore code come QM1. Il gestore code viene personalizzato in base ai requisiti dell'utente, ad esempio il nome del gestore code originale.

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

Il codice precedente estrae la configurazione del gestore code importata nella sezione Importa la configurazione IBM MQ . Se sono stati utilizzati nomi diversi, è necessario modificare `my-mqsc-migrated` e `backup.mqsc`.

Si noti che lo YAML di esempio presuppone che la classe di memoria predefinita per l'ambiente Red Hat OpenShift sia definita come una classe di memoria RWX o RWO. Se non è definito un valore predefinito nel proprio ambiente, è necessario specificare la classe di memoria da utilizzare. È possibile eseguire questa operazione estendendo YAML nel modo seguente:

```
  queueManager:
    name: QM1
    storage:
      defaultClass: my_storage_class
```

```
      queueManager:
        type: persistent-claim
```

Aggiungere il testo evidenziato, con l'attributo della classe personalizzato per corrispondere al proprio ambiente. Per rilevare i nomi delle classi di memoria nell'ambiente, eseguire il seguente comando:

```
oc get storageclass
```

Di seguito viene riportato un output di esempio restituito da questo comando:

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

Il seguente codice mostra come fare riferimento alla configurazione IBM MQ importata nella sezione Importa la configurazione IBM MQ . Se sono stati utilizzati nomi diversi, è necessario modificare `my-mqsc-migrated` e `backup.mqsc`.

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

Il gestore code a istanza singola è stato distribuito. Questo completa il modello. Sei ora pronto a verificare la distribuzione del nuovo contenitore.

- Distribuire un gestore code a più istanze.

  Il gestore code migrato viene distribuito a Red Hat OpenShift utilizzando un file YAML. Il seguente esempio è basato sui nomi utilizzati nelle precedenti sezioni.

```
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
```

Ecco una spiegazione di questo YAML. La maggior parte della configurazione segue lo stesso approccio di distribuzione di un singolo gestore code dell'istanza, pertanto vengono illustrati solo gli aspetti relativi alla disponibilità e alla memoria del gestore code.

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

Specifica il nome del gestore code come `QM1`e imposta la distribuzione su `MultiInstance` invece che sulla singola istanza predefinita.

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

Un gestore code a più istanze IBM MQ dipende dalla memoria RWX. Per impostazione predefinita, un gestore code viene distribuito in modalità a istanza singola e, pertanto, sono richieste ulteriori opzioni di archiviazione quando si passa alla modalità a più istanze. Nel precedente esempio YAML, vengono definiti tre volumi di archiviazione persistenti e una classe di volume persistente. Questa classe di volume persistente deve essere una classe di archiviazione RWX. Se non si è sicuri dei nomi delle classi di memoria nel proprio ambiente, è possibile eseguire il seguente comando per rilevarli:

```
oc get storageclass
```

Di seguito viene riportato un output di esempio restituito da questo comando:

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

Il seguente codice mostra come fare riferimento alla configurazione IBM MQ importata nella sezione Importa la configurazione IBM MQ . Se sono stati utilizzati nomi diversi, è necessario modificare `my-mqsc-migrated` e `backup.mqsc`.

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

È stato distribuito il gestore code a più istanze. Questo completa il modello. Sei ora pronto a verificare la distribuzione del nuovo contenitore.

OpenShift V 9.2.1 EUS CD Verifica della nuova distribuzione del contenitore

Ora che IBM MQ è distribuito su Red Hat OpenShift, puoi verificare l'ambiente utilizzando gli esempi IBM MQ .

Prima di iniziare

Questa attività presuppone che sia stato creato il nuovo gestore code su Red Hat OpenShift.

Importante: Questa attività presuppone che TLS non sia abilitato nel gestore code.

Informazioni su questa attività

In questa attività si eseguono gli esempi IBM MQ dall'interno del contenitore del gestore code migrato. Tuttavia, è possibile utilizzare le proprie applicazioni in esecuzione da un altro ambiente.

Hai bisogno delle seguenti informazioni:

- Nome utente LDAP
- Password LDAP
- IBM MQ Nome canale
- Nome coda

Questo codice di esempio usa le seguenti impostazioni. Si prega di notare che le impostazioni saranno diverse.

- Nome utente LDAP: mqapp
- Password LDAP: mqapp
- IBM MQ Nome canale: DEV.APP.SVRCONN
- Nome coda: Q1

Procedura

1. Esegui nel contenitore IBM MQ in esecuzione.

   Utilizzare il seguente comando:

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

   dove `qm1-ibm-mq-0` è il pod che abbiamo distribuito in “Creazione del nuovo gestore code su Red Hat OpenShift” a pagina 53. Se la distribuzione è stata richiamata in modo diverso, personalizzare questo valore.
2. Inviare un messaggio.

   Eseguire i seguenti comandi:

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

   Viene richiesta una password, quindi è possibile inviare un messaggio.
3. Verificare che il messaggio sia stato ricevuto correttamente.

   Eseguire l'esempio GET:

   ```
   ./amqsgetc Q1 QM1
   ```

Risultati

È stato completato il “Migrazione di IBM MQ a IBM Cloud Pak for Integration” a pagina 36.

Operazioni successive

Utilizzare le seguenti informazioni come supporto per scenari di migrazione più complessi:

Migrazione dei messaggi accodati
Per migrare i messaggi in coda esistenti, seguire le istruzioni riportate nel seguente argomento per l'esportazione e l'importazione dei messaggi dopo che il nuovo gestore code è stato creato: Utilizzo del programma di utilità dmpmqmsg tra due sistemi.

Connessione a IBM MQ dall'esterno dell'ambiente Red Hat OpenShift

Il gestore code distribuito può essere esposto ai client IBM MQ e ai gestori code all'esterno dell'ambiente Red Hat OpenShift . Il processo dipende dalla versione di IBM MQ che si collega all'ambiente Red Hat OpenShift . Vedere “Configurazione di un instradamento per la connessione a un gestore code dall'esterno di un cluster Red Hat OpenShift” a pagina 108.

OpenShift CP4I Installazione e disinstallazione di IBM MQ Operator su Red Hat OpenShift

Il IBM MQ Operator può essere installato su Red Hat OpenShift utilizzando l'hub operatore.

Procedura

- “Dipendenze per IBM MQ Operator” a pagina 10.
- “Autorizzazioni nell'ambito del cluster richieste da IBM MQ Operator” a pagina 11.
- “Installazione di IBM MQ Operator utilizzando la console web Red Hat OpenShift” a pagina 59.
- “Installazione di IBM MQ Operator utilizzando la CLI di Red Hat OpenShift” a pagina 61.
- “Installazione di IBM MQ Operator in un ambiente airgap” a pagina 65.

Attività correlate

“Disinstallazione di IBM MQ Operator utilizzando la console web Red Hat OpenShift” a pagina 61
È possibile utilizzare la console web Red Hat OpenShift per disinstallare IBM MQ Operator da Red Hat OpenShift.

“Disinstallazione di IBM MQ Operator utilizzando la CLI Red Hat OpenShift” a pagina 64
Puoi utilizzare la CLI Red Hat OpenShift per disinstallare IBM MQ Operator da Red Hat OpenShift. Esistono differenze nel processo di disinstallazione, a seconda che IBM MQ Operator sia installato in un singolo spazio dei nomi o installato e disponibile per tutti gli spazi dei nomi sul cluster.

OpenShift CP4I Installazione di IBM MQ Operator utilizzando la console web Red Hat OpenShift

Il IBM MQ Operator può essere installato su Red Hat OpenShift utilizzando l'hub operatore.

Prima di iniziare

Accedi alla tua console web del cluster Red Hat OpenShift .

Procedura

1. EUS

   Opzionale: Aggiungere gli operatori IBM Common Services all'elenco di operatori installabili.

   Nota:

   Questa operazione si applica alle release di IBM MQ Operator 1.5 e precedenti. Il passo aggiunge un catalogo Common Services separato. Per le release successive dell'operatore, i Common Services sono inclusi nel catalogo IBM .

   a) Fare clic sull'icona più in alto a destra dello schermo. Viene visualizzata la finestra di dialogo **Importa YAM** .

   b) Incollare la seguente definizione risorsa nella casella di dialogo.

```
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) Fai clic su **Crea**.

2. Aggiungere gli operatori IBM all'elenco di operatori installabili

a) Fare clic sull'icona più in alto a destra dello schermo. Viene visualizzata la finestra di dialogo **Importa YAM** .

b) Incollare la seguente definizione risorsa nella casella di dialogo.

```
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) Fai clic su **Crea**.

3. Creare uno spazio dei nomi da utilizzare per IBM MQ Operator

Il IBM MQ Operator può essere installato nell'ambito di un singolo spazio dei nomi o di tutti gli spazi dei nomi. Questa fase è necessaria solo se si desidera eseguire l'installazione in uno spazio dei nomi particolare che non esiste già.

a) Dal pannello di navigazione, fare clic su **Home** > **Progetti**.

Viene visualizzata la pagina Progetti.

b) Fare clic su **Crea progetto**. Viene visualizzata un'area Crea progetto.

c) Immetti i dettagli dello spazio dei nomi che stai creando. Ad esempio, è possibile specificare "ibm - mq" come nome.

d) Fai clic su **Crea**. Lo spazio dei nomi per il tuo IBM MQ Operator viene creato.

4. Installare IBM MQ Operator.

a) Dal riquadro di navigazione, fare clic su **Operatori** > **OperatorHub**.

Viene visualizzata la pagina OperatorHub .

b) Nel campo **Tutti gli elementi** , immettere "IBM MQ".

Viene visualizzata la voce di catalogo IBM MQ .

c) Selezionare **IBM MQ**.

Viene visualizzata la finestra IBM MQ .

d) Fai clic su **Installa**.

Viene visualizzata la pagina Crea sottoscrizione operatore.

e) Consultare “Supporto versione per IBM MQ Operator” a pagina 7 per stabilire quale canale operatore scegliere.

f) Impostare la modalità di installazione sullo spazio dei nomi specifico creato o sull'ambito a livello di cluster.

Si consiglia di scegliere l'ambito a livello di cluster, poiché l'installazione di versioni differenti di un operatore in spazi dei nomi differenti può causare problemi. Gli operatori sono progettati per essere estensioni del piano di controllo.

g) Fai clic su **Sottoscrivi**.

Verrà visualizzato IBM MQ nella pagina Operatori installati.

h) Controllare lo stato dell'operatore nella pagina Operatori installati; lo stato cambierà in Riuscito quando l'installazione è completa.

Operazioni successive

"Preparazione del tuo progetto Red Hat OpenShift per IBM MQ utilizzando la console Web Red Hat OpenShift" a pagina 80

OpenShift CP4I *Disinstallazione di IBM MQ Operator utilizzando la console web Red Hat OpenShift*

È possibile utilizzare la console web Red Hat OpenShift per disinstallare IBM MQ Operator da Red Hat OpenShift.

Prima di iniziare

Accedi alla console web del tuo cluster Red Hat OpenShift .

Se IBM MQ Operator è installato su tutti i progetti / spazi dei nomi sul cluster, ripetere i passi da 1 a 5 della seguente procedura per ogni progetto in cui si desidera eliminare i gestori code.

Procedura

1. Selezionare **Operatori** > **Operatori installati**.
2. Dall'elenco a discesa **Progetto** , selezionare un progetto.
3. Fare clic sull'operatore **IBM MQ** .
4. Selezionare la scheda **Gestori code** per visualizzare i gestori code gestiti da questo IBM MQ Operator.
5. Eliminare uno o più gestori code.

   Si noti che, sebbene questi gestori code continuino ad essere in esecuzione, potrebbero non funzionare come previsto senza un IBM MQ Operator.
6. Opzionale: Se appropriato, ripetere i passi da 1 a 5 per ogni progetto in cui si desidera eliminare i gestori code.
7. Tornare a **Operatori** > **Operatori installati**.
8. Accanto all'operatore **IBM MQ** , fare clic sul menu a tre punti e selezionare **Disinstalla operatore**.
9. Se si utilizza Red Hat OpenShift Container Platform 4.7, potrebbe essere necessario eliminare manualmente l'hook web di convalida dalla riga comandi:

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

OpenShift CP4I Installazione di IBM MQ Operator utilizzando la CLI di Red Hat OpenShift

Il IBM MQ Operator può essere installato su Red Hat OpenShift utilizzando l'hub operatore.

Prima di iniziare

Accedi alla CLI (command line interface) Red Hat OpenShift utilizzando **`oc login`**. Per questa procedura, sarà necessario essere un amministratore del cluster.

Procedura

1. EUS

   Opzionale: Creare un **`CatalogSource`** per gli operatori IBM Common Services .

   Nota:

Questa operazione si applica alle release di IBM MQ Operator 1.5 e precedenti. Il passo aggiunge un catalogo Common Services separato. Per le release successive dell'operatore, i Common Services sono inclusi nel catalogo IBM .

a) Creare un file YAML definendo la risorsa **CatalogSource** .

Creare un file denominato "operator-source-cs.yaml" con il contenuto seguente:

```
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) Applicare il **CatalogSource** al server.

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

2. Crea un **CatalogSource** per gli operatori IBM

a) Creare un file YAML che definisca la risorsa **CatalogSource**

Creare un file denominato "operator-source-ibm.yaml" con il contenuto seguente:

```
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) Applicare il **CatalogSource** al server.

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

3. Creare uno spazio dei nomi da utilizzare per IBM MQ Operator

Il IBM MQ Operator può essere installato nell'ambito di un singolo spazio dei nomi o di tutti gli spazi dei nomi. Questa fase è necessaria solo se si desidera eseguire l'installazione in uno spazio dei nomi particolare che non esiste già.

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

4. Visualizzare l'elenco di operatori disponibili per il cluster da OperatorHub

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

5. Esaminare IBM MQ Operator per verificare le InstallModes supportate e i canali disponibili

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

6. Crea un file YAML dell'oggetto **OperatorGroup**

Una **OperatorGroup** è una risorsa OLM che seleziona gli spazi dei nomi di destinazione in cui generare l'accesso RBAC richiesto per tutti gli operatori nello stesso spazio dei nomi di **OperatorGroup**.

Lo spazio dei nomi a cui sottoscrivi l'operatore deve avere un **OperatorGroup** che corrisponde alla modalità **InstallMode**dell'operatore, `AllNamespaces` o `SingleNamespace` . Se l'operatore che si intende installare utilizza `AllNamespaces`, lo spazio dei nomi `openshift-operators` ha già un **OperatorGroup** appropriato.

Tuttavia, se l'operatore utilizza la modalità `SingleNamespace` e non si dispone già di un **OperatorGroup** appropriato, è necessario crearne uno.

a) Creare un file denominato "mq-operator-group.yaml" con il seguente contenuto:

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

b) Creare l'oggetto **OperatorGroup**

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

7. Creare un file YAML dell'oggetto **Subscription** per sottoscrivere uno spazio dei nomi in IBM MQ Operator

a) Consultare “Supporto versione per IBM MQ Operator” a pagina 7 per stabilire quale canale operatore scegliere.

b) Crea un file denominatomq-sub.yaml" con i seguenti contenuti, ma modificando **channel** in modo che corrisponda al canale per la versione del IBM MQ Operator che desideri installare.

```
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
```

Per l'utilizzo `AllNamespaces` **InstallMode** , specificare **openshift-operators** nello spazio dei nomi. Altrimenti, specificare il singolo spazio dei nomi pertinente per l'utilizzo di `SingleNamespace` **InstallMode** . Notare che è necessario modificare solo il campo **namespace** , lasciando invariato il campo **sourceNamespace** .

c) Creare l'oggetto **Subscription**

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

8. Verificare lo stato dell'operatore

Una volta eseguita correttamente l'installazione dell'operatore, lo stato del pod viene visualizzato come *In esecuzione*. Per l'utilizzo `AllNamespaces` **InstallMode** , specificare **openshift-operators** come spazio dei nomi. Altrimenti, specificare il singolo spazio dei nomi pertinente per l'utilizzo di `SingleNamespace` **InstallMode** .

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

Operazioni successive

“Preparazione del tuo progetto di Red Hat OpenShift per IBM MQ utilizzando la CLI Red Hat OpenShift” a pagina 80

OpenShift CP4I *Disinstallazione di IBM MQ Operator utilizzando la CLI Red Hat OpenShift*

Puoi utilizzare la CLI Red Hat OpenShift per disinstallare IBM MQ Operator da Red Hat OpenShift. Esistono differenze nel processo di disinstallazione, a seconda che IBM MQ Operator sia installato in un singolo spazio dei nomi o installato e disponibile per tutti gli spazi dei nomi sul cluster.

Prima di iniziare

Accedi al tuo cluster Red Hat OpenShift utilizzando `oc login`.

Procedura

- Se IBM MQ Operator è installato in un singolo spazio dei nomi, completare i seguenti passi secondari:
  a) Assicurarsi di essere nel progetto corretto:

  ```
  oc project <project_name>
  ```

  b) Visualizzare i gestori code installati nel progetto:

  ```
  oc get qmgr
  ```

  c) Eliminare uno o più gestori code:

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

  Si noti che, sebbene questi gestori code continuino ad essere in esecuzione, potrebbero non funzionare come previsto senza un IBM MQ Operator.

  d) Visualizzare le istanze **`ClusterServiceVersion`** :

  ```
  oc get csv
  ```

  e) Eliminare il IBM MQ **`ClusterServiceVersion`**:

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

  f) Visualizzare le sottoscrizioni:

  ```
  oc get subscription
  ```

  g) Elimina tutte le sottoscrizioni:

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

  h) Opzionale: Se nessun altro utilizza i servizi comuni, è possibile disinstallare l'operatore dei servizi comuni ed eliminare il gruppo di operatori:

  a. Disinstallare l'operatore dei servizi comuni, seguendo le istruzioni in Disinstallazione dei servizi comuni nella documentazione di IBM Cloud Pak foundational services .

  b. Visualizzare il gruppo di operatori:

  ```
  oc get operatorgroup
  ```

  c. Eliminare il gruppo di operatori:

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

- Se il IBM MQ Operator è installato e disponibile per tutti gli spazi dei nomi sul cluster, completare i seguenti passi secondari:
  a) Visualizzare tutti i gestori code installati:

  ```
  oc get qmgr -A
  ```

b) Eliminare uno o più gestori code:

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

Si noti che, sebbene questi gestori code continuino ad essere in esecuzione, potrebbero non funzionare come previsto senza un IBM MQ Operator.

c) Visualizzare le istanze **ClusterServiceVersion** :

```
oc get csv -A
```

d) Eliminare il IBM MQ **ClusterServiceVersion** dal cluster:

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

e) Visualizzare le sottoscrizioni:

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

f) Eliminare le sottoscrizioni:

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

g) Se si utilizza Red Hat OpenShift Container Platform 4.7, potrebbe essere necessario eliminare manualmente l'hook web di convalida:

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

h) Opzionale: Se nessun altro utilizza i servizi comuni, è possibile disinstallare l'operatore dei servizi comuni:

Seguire le istruzioni in Disinstallazione dei servizi comuni nella documentazione del prodotto IBM Cloud Pak foundational services .

Installazione di IBM MQ Operator in un ambiente airgap

Questa esercitazione ti guida nell'installazione di IBM MQ Operator in un cluster Red Hat OpenShift che non ha connettività internet. Puoi installare IBM MQ Operator in un ambiente airgap utilizzando un dispositivo di archiviazione portatile o utilizzando una macchina bastion.

Installazione di IBM MQ Operator in un ambiente airgap utilizzando un dispositivo di memorizzazione portatile

Per la procedura di completamento dell'installazione, consultare Mirroring delle immagini con una periferica di archiviazione portatile nella documentazione di IBM Cloud Pak for Integration . Se si sta installando solo IBM MQ, sostituire tutte le ricorrenze delle seguenti variabili di ambiente con i valori forniti di seguito:

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

dove *version_number* è la versione del caso che si desidera utilizzare per eseguire l'installazione airgap. Per un elenco delle versioni del caso disponibili, vedere https://github.com/IBM/cloud-pak/tree/master/repo/case/ibm-mq. Consultare “Supporto versione per IBM MQ Operator” a pagina 7 per stabilire quale canale operatore scegliere.

Installazione di IBM MQ Operator in un ambiente airgap utilizzando una macchina bastion

1. “Prerequisiti” a pagina 66

2. “Preparare un registro Docker” a pagina 66
3. “Prepara un host bastion” a pagina 67
4. “Crea variabili di ambiente per il programma di installazione e l'inventario immagini” a pagina 68
5. “Scarica il programma di installazione IBM MQ e l'inventario delle immagini” a pagina 68
6. “Accedere al cluster Red Hat OpenShift Container Platform come amministratore del cluster” a pagina 68
7. “Crea uno spazio dei nomi Kubernetes per IBM MQ Operator” a pagina 68
8. “Eseguire il mirroring delle immagini e configurare il cluster” a pagina 68
9. “Installare IBM MQ Operator.” a pagina 70
10. “Distribuisci gestore code IBM MQ” a pagina 71

Prerequisiti

1. È necessario installare un cluster di Red Hat OpenShift Container Platform . Per le versioni Red Hat OpenShift Container Platform supportate, consultare “Supporto versione per IBM MQ Operator” a pagina 7.
2. Deve essere disponibile un registro Docker . Per ulteriori informazioni, consultare “Preparare un registro Docker” a pagina 66.
3. È necessario configurare un bastion server. Per ulteriori informazioni, consultare “Prepara un host bastion” a pagina 67.

Preparare un registro Docker

Un registro Docker locale viene utilizzato per memorizzare tutte le immagini nel tuo ambiente locale. È necessario creare tale registro e assicurarsi che soddisfi i seguenti requisiti:

- Supporta Docker Manifest V2, Schema 2.
- Supporta immagini multi - architettura.
- È accessibile sia dal server bastion che dai tuoi nodi cluster Red Hat OpenShift Container Platform .
- Dispone del nome utente e della parola d'ordine di un utente che può scrivere nel registro di destinazione dall'host bastion.
- Ha il nome utente e la password di un utente che può leggere dal registro di destinazione che si trova sui nodi cluster Red Hat OpenShift .
- Consente i separatori di percorso nel nome immagine.

Dopo aver creato il Registro di sistema di Docker , è necessario configurare il registro:

1. Crea spazi dei nomi del registro
   - `ibmcom` - Spazio dei nomi per memorizzare tutte le immagini dallo spazio dei nomi `dockerhub.io/ibmcom` .

     Lo spazio dei nomi `ibmcom` è per tutte le immagini IBM che sono pubblicamente disponibili e non richiedono credenziali per il pull.
   - `cp` - Namespace per archiviare le immagini IBM dal repository `cp.icr.io/cp` .

     Lo spazio dei nomi `cp` è per le immagini in IBM Entitled Registry che richiedono una chiave di titolarità del prodotto e le credenziali per il pull. Per ottenere la tua chiave di titolarità, accedi a MyIBM Container Software Library con ID e password IBM associati al software autorizzato. Nella sezione **Chiavi di titolarità** , selezionare **Copia chiave** per copiare la chiave di titolarità negli appunti, quindi salvarla per utilizzarla nei seguenti passi.
   - `opencloudio` - Namespace per memorizzare le immagini da `quay.io/opencloudio`.

Lo spazio dei nomi `opencloudio` è per selezionare IBM immagini del componente open source disponibili su quay.io. Le immagini IBM Cloud Pak foundational services sono ospitate su `opencloudio`.

2. Verificare che ogni spazio dei nomi soddisfi i seguenti requisiti:
   - Supporta la creazione automatica del repository.
   - Dispone delle credenziali di un utente che può scrivere e creare repository. L'host bastion utilizza queste credenziali.
   - Dispone delle credenziali di un utente che può leggere tutti i repository. Il cluster Red Hat OpenShift Container Platform utilizza queste credenziali.

Prepara un host bastion

Prepara un host bastion che può accedere al cluster Red Hat OpenShift Container Platform , al registro Docker locale e a Internet. L'host bastion deve essere su una piattaforma Linux for x86-64 con qualsiasi sistema operativo supportato dalla CLI IBM Cloud Pak e dalla CLI Red Hat OpenShift Container Platform .

Completa questa procedura sul tuo nodo bastion:

1. Installare OpenSSL versione 1.11.1 o successiva.
2. Installa Docker o Podman sul nodo bastion.
   - Per installare Docker, eseguire questi comandi:

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

   - Per installare Podman, vedi Podman Installation Instructions
3. Installa skopeo versione 1.x.x sul nodo bastion. Per installare skopeo, eseguire questi comandi:

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

4. Installa la CLI IBM Cloud Pak. Installare la versione più recente del file binario per la piattaforma. Per ulteriori informazioni, vedi cloud - pak - cli.
   a. Scaricare il file binario.

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

      Ad esempio:

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

   b. Estrarre il file binario.

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

   c. Utilizzare i seguenti comandi per modificare e spostare il file

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

   d. Confermare che `cloudctl` sia installato:

      ```
      cloudctl --help
      ```

5. Installa lo strumento CLI `oc` Red Hat OpenShift Container Platform .

   Per ulteriori informazioni, vedi Strumenti CLIRed Hat OpenShift Container Platform
6. Creare una directory che funga da negozio offline.

La seguente è una directory di esempio. Questo esempio viene utilizzato nelle fasi successive.

```
mkdir $HOME/offline
```

Nota: questo negozio non in linea deve essere persistente per evitare il trasferimento dei dati più di una volta. La persistenza consente inoltre di eseguire il processo di mirroring più volte o in base a una pianificazione.

Crea variabili di ambiente per il programma di installazione e l'inventario immagini

Creare le seguenti variabili di ambiente con il nome immagine del programma di installazione e l'inventario immagini:

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

dove *version_number* è la versione del caso che si desidera utilizzare per eseguire l'installazione airgap. Per un elenco delle versioni del caso disponibili, vedere https://github.com/IBM/cloud-pak/tree/master/repo/case/ibm-mq. Esaminare il supporto della versione per IBM MQ Operator per stabilire quale canale operatore scegliere.

Scarica il programma di installazione IBM MQ e l'inventario delle immagini

Scarica l'inventario di immagini e del programma di installazione `ibm-mq` sull'host 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/
```

Accedere al cluster Red Hat OpenShift Container Platform come amministratore del cluster

Il seguente è un comando di esempio per accedere al cluster Red Hat OpenShift Container Platform :

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

Crea uno spazio dei nomi Kubernetes per IBM MQ Operator

Crea una variabile di ambiente con uno spazio dei nomi per installare IBM MQ Operator, quindi crea lo spazio dei nomi:

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

Eseguire il mirroring delle immagini e configurare il cluster

Completa questa procedura per eseguire il mirroring delle immagini e configurare il tuo cluster:

Nota: Non utilizzare la tilde tra virgolette doppie in alcun comando. Ad esempio, non utilizzare `args "--registry registry --user registry_userid --pass registry_password --inputDir ~/offline"`. La tilde non si espande e i comandi potrebbero avere esito negativo.

1. Memorizzare le credenziali di autenticazione per tutti i registri Docker di origine.

   Tutte le immagini IBM Cloud Platform Common Services, IBM MQ Operator e IBM MQ Advanced Developer sono archiviate in registri pubblici che non richiedono l'autenticazione. Tuttavia, IBM MQ Advanced Server (non sviluppatore), altri prodotti e componenti di terze parti richiedono uno o più registri autenticati. I seguenti registri richiedono l'autenticazione:

   - `cp.icr.io`

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

Per ulteriori informazioni su questi registri, vedi Crea spazi dei nomi del registro.

Devi eseguire il seguente comando per configurare le credenziali per tutti i registri che richiedono autenticazione. Eseguire il comando separatamente per ciascun registro:

```
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"
```

Il comando memorizza e memorizza nella cache le credenziali del Registro di sistema in un file sul file system nell'ubicazione $HOME/`.airgap/secrets` .

2. Creare le variabili di ambiente con le informazioni di connessione del registro Docker locale.

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

Nota: il registro di Docker utilizza porte standard come 80 o 443. Se il registro Docker utilizza una porta non standard, specificare la porta utilizzando la sintassi *host*:*port*. Ad esempio:

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

3. Configurare un segreto di autenticazione per il registro Docker locale.

Nota: questa operazione deve essere eseguita una sola volta.

```
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}"
```

Il comando memorizza e memorizza nella cache le credenziali del Registro di sistema in un file sul file system nell'ubicazione $HOME/`.airgap/secrets` .

4. Configurare un segreto di pull dell'immagine globale e **ImageContentSourcePolicy**.

   a. Verificare se è richiesto un riavvio del nodo.

   - In Red Hat OpenShift Container Platform versione 4.4 e successive, e in una nuova installazione di IBM MQ Operator utilizzando airgap, questo passaggio riavvia tutti i nodi cluster. Le risorse del cluster potrebbero non essere disponibili fino a quando non viene applicato il nuovo segreto di pull.
   - In IBM MQ Operator 1.8, CASE viene aggiornato per includere un'altra origine di mirroring per le immagini. Pertanto, quando si esegue l'aggiornamento da versioni precedenti di IBM MQ Operator alla versione 1.8 o successive, viene attivato un riavvio del nodo.
   - Per verificare se questo passo richiede un riavvio del nodo, aggiungere l'opzione `--dry - run` al codice per questo passo. Questo genera l'ultima **ImageContentSourcePolicy** e la visualizza nella finestra della console (**stdout**). Se questo **ImageContentSourcePolicy** è diverso dal cluster configurato **ImageContentSourcePolicy**, si verifica un riavvio.

     ```
     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. Per configurare il segreto di pull dell'immagine globale e **ImageContentSourcePolicy**, esegui il codice per questo passo senza l'opzione `--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. Verificare che la risorsa **ImageContentSourcePolicy** sia stata creata.

```
oc get imageContentSourcePolicy
```

6. Facoltativo: se si utilizza un registro non sicuro, è necessario aggiungere il registro locale all'elenco **insecureRegistries** del cluster.

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

7. Verificare lo stato del nodo cluster.

```
oc get nodes
```

Una volta applicati il segreto di pull dell'immagine globale e **imageContentsourcePolicy** , potresti visualizzare lo stato del nodo come **Ready**, **Scheduling**o **Disabled**. Attendere che tutti i nodi mostrino uno stato **Ready** .

8. Eseguire il mirroring delle immagini nel registro locale.

```
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"
```

Installare IBM MQ Operator.

1. Accedi alla tua console web del cluster Red Hat OpenShift .
2. Creare un'origine catalogo. Utilizzare lo stesso terminale che ha eseguito i passi precedenti.

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

3. Verificare che **CatalogSource** sia creato per l'operatore del programma di installazione di Common Services .

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

4. Installare IBM MQ Operator utilizzando OLM.
   a. Dal riquadro di navigazione, fare clic su **Operatori** > **OperatorHub**.

      Viene visualizzata la pagina **OperatorHub** .

   b. Nel campo **Tutti gli elementi** , immettere `IBM MQ`.

      Viene visualizzata la voce di catalogo IBM MQ .

   c. Selezionare **IBM MQ**.

      Viene visualizzata la finestra **IBM MQ** .

d. Fai clic su **Installa**.

   Viene visualizzata la pagina **Crea sottoscrizione operatore** .

e. Consultare “Supporto versione per IBM MQ Operator” a pagina 7 per stabilire quale canale operatore scegliere.

f. Impostare la **Modalità di installazione** sullo spazio dei nomi specifico creato o sull'ambito del cluster.

g. Fai clic su **Sottoscrivi**.

   IBM MQ viene aggiunto alla pagina **Operatori installati** .

h. Verificare lo stato dell'operatore nella pagina **Operatori installati** . Lo stato cambia in **Succeeded** quando l'installazione è completa.

Distribuisci gestore code IBM MQ

Per creare un nuovo gestore code sotto l'operatore installato, consultare “Distribuzione e configurazione dei gestori code utilizzando IBM MQ Operator” a pagina 79.

Attività correlate

“Preparazione all'aggiornamento del IBM MQ Operator o del gestore code in un ambiente airgap” a pagina 71
In un cluster Red Hat OpenShift che non dispone di connettività Internet, ci sono delle fasi preparatorie che devi eseguire prima di aggiornare IBM MQ Operator.

OpenShift CP4I Aggiornamento di IBM MQ Operator e dei gestori code

L'aggiornamento di IBM MQ Operator consente di aggiornare i gestori code.

Procedura

- “Aggiornamento di IBM MQ Operator utilizzando la Console web di Red Hat OpenShift” a pagina 74.
- “Aggiornamento di IBM MQ Operator utilizzando la CLI di Red Hat OpenShift” a pagina 76.
- “Aggiornamento di un gestore code IBM MQ utilizzando la console Red Hat OpenShift web” a pagina 77.
- “Aggiornamento di un gestore code IBM MQ utilizzando la CLI Red Hat OpenShift” a pagina 78.

OpenShift CP4I Linux Preparazione all'aggiornamento del IBM MQ Operator o del gestore code in un ambiente airgap

In un cluster Red Hat OpenShift che non dispone di connettività Internet, ci sono delle fasi preparatorie che devi eseguire prima di aggiornare IBM MQ Operator.

Prima di iniziare

Questo argomento presuppone che sia già stato configurato un registro di immagini locale in cui vengono sottoposte a mirroring le immagini IBM Cloud Pak for Integration rilasciate in precedenza.

Informazioni su questa attività

Prima di poter aggiornare IBM MQ Operator o il gestore code in un ambiente airgap, è necessario eseguire il mirroring delle immagini IBM Cloud Pak for Integration più recenti.

Notare che i primi quattro passi in questa attività sono gli stessi che si eseguono quando “Installazione di IBM MQ Operator in un ambiente airgap” a pagina 65.

Procedura

1. Creare le variabili di ambiente per il programma di installazione e l'inventario immagini.

   Creare le seguenti variabili di ambiente con il nome immagine del programma di installazione e l'inventario immagini:

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

   dove *version_number* è la versione del caso che si desidera utilizzare per eseguire l'installazione airgap. Per un elenco delle versioni del caso disponibili, vedere https://github.com/IBM/cloud-pak/tree/master/repo/case/ibm-mq. Esaminare il supporto della versione per IBM MQ Operator per stabilire quale canale operatore scegliere.
2. Scaricare il programma di installazione IBM MQ e l'inventario delle immagini.

   Scarica l'inventario di immagini e del programma di installazione `ibm-mq` sull'host 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. Accedere al cluster Red Hat OpenShift Container Platform come amministratore del cluster.

   Il seguente è un comando di esempio per accedere al cluster Red Hat OpenShift Container Platform :

   ```
   oc login cluster_host:port --username=cluster_admin_user --password=cluster_admin_password
   ```
4. Eseguire il mirroring delle immagini e configurare il cluster.

   Completa questa procedura per eseguire il mirroring delle immagini e configurare il tuo cluster:

   Nota: Non utilizzare la tilde tra virgolette doppie in alcun comando. Ad esempio, non utilizzare `args "--registry registry --user registry_userid --pass registry_password --inputDir ~/offline"`. La tilde non si espande e i comandi potrebbero avere esito negativo.

   a. Memorizzare le credenziali di autenticazione per tutti i registri Docker di origine.

      Tutte le immagini IBM Cloud Platform Common Services, IBM MQ Operator e IBM MQ Advanced Developer sono archiviate in registri pubblici che non richiedono l'autenticazione. Tuttavia, IBM MQ Advanced Server (non sviluppatore), altri prodotti e componenti di terze parti richiedono uno o più registri autenticati. I seguenti registri richiedono l'autenticazione:

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

      Per ulteriori informazioni su questi registri, vedi Crea spazi dei nomi del registro.

      Devi eseguire il seguente comando per configurare le credenziali per tutti i registri che richiedono autenticazione. Eseguire il comando separatamente per ciascun registro:

      ```
      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"
      ```

      Il comando memorizza e memorizza nella cache le credenziali del Registro di sistema in un file sul file system nell'ubicazione `$HOME/.airgap/secrets` .

b. Creare le variabili di ambiente con le informazioni di connessione del registro Docker locale.

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

Nota: il registro di Docker utilizza porte standard come 80 o 443. Se il registro Docker utilizza una porta non standard, specificare la porta utilizzando la sintassi *host:port*. Ad esempio:

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

c. Configurare un segreto di autenticazione per il registro Docker locale.

Nota: questa operazione deve essere eseguita una sola volta.

```
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}"
```

Il comando memorizza e memorizza nella cache le credenziali del Registro di sistema in un file sul file system nell'ubicazione $HOME/.airgap/secrets .

d. Configurare un segreto di pull dell'immagine globale e **ImageContentSourcePolicy**.

i) Verificare se è richiesto un riavvio del nodo.

- In Red Hat OpenShift Container Platform versione 4.4 e successive, e in una nuova installazione di IBM MQ Operator utilizzando airgap, questo passaggio riavvia tutti i nodi cluster. Le risorse del cluster potrebbero non essere disponibili fino a quando non viene applicato il nuovo segreto di pull.
- In IBM MQ Operator 1.8, CASE viene aggiornato per includere un'altra origine di mirroring per le immagini. Pertanto, quando si esegue l'aggiornamento da versioni precedenti di IBM MQ Operator alla versione 1.8 o successive, viene attivato un riavvio del nodo.
- Per verificare se questo passo richiede un riavvio del nodo, aggiungere l'opzione `--dry -run` al codice per questo passo. Questo genera l'ultima **ImageContentSourcePolicy** e la visualizza nella finestra della console (**stdout**). Se questo **ImageContentSourcePolicy** è diverso dal cluster configurato **ImageContentSourcePolicy**, si verifica un riavvio.

```
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) Per configurare il segreto di pull dell'immagine globale e **ImageContentSourcePolicy**, esegui il codice per questo passo senza l'opzione `--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. Verificare che la risorsa **ImageContentSourcePolicy** sia stata creata.

```
oc get imageContentSourcePolicy
```

f. Facoltativo: se si utilizza un registro non sicuro, è necessario aggiungere il registro locale all'elenco **insecureRegistries** del cluster.

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

g. Verificare lo stato del nodo cluster.

```
oc get nodes
```

Una volta applicati il segreto di pull dell'immagine globale e **imageContentsourcePolicy** , potresti visualizzare lo stato del nodo come **Ready**, **Scheduling**o **Disabled**. Attendere che tutti i nodi mostrino uno stato **Ready** .

h. Eseguire il mirroring delle immagini nel registro locale.

```
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. Aggiornare l'origine del catalogo.

Utilizzare lo stesso terminale che ha eseguito i passi precedenti.

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

Operazioni successive

È ora possibile aggiornare IBM MQ Operator e il gestore code, completando una delle seguenti attività:

- “Aggiornamento di IBM MQ Operator utilizzando la Console web di Red Hat OpenShift” a pagina 74
- “Aggiornamento di IBM MQ Operator utilizzando la CLI di Red Hat OpenShift” a pagina 76
- “Aggiornamento di un gestore code IBM MQ utilizzando la console Red Hat OpenShift web” a pagina 77
- “Aggiornamento di un gestore code IBM MQ utilizzando la CLI Red Hat OpenShift” a pagina 78
- “Aggiornamento di un gestore code IBM MQ in Red Hat OpenShift utilizzando la piattaforma Navigator” a pagina 78

OpenShift CP4I Aggiornamento di IBM MQ Operator utilizzando la Console web di Red Hat OpenShift

Il IBM MQ Operator può essere aggiornato utilizzando l'hub dell'operatore.

Prima di iniziare

Accedi alla tua console web del cluster Red Hat OpenShift .

Prima di poter aggiornare il IBM MQ Operator in un ambiente airgap, è necessario eseguire il mirroring delle immagini IBM Cloud Pak for Integration più recenti. Consultare Preparazione per l'aggiornamento del IBM MQ Operator o del gestore code in un ambiente airgap.

Procedura

1. Esaminare “Supporto versione per IBM MQ Operator” a pagina 7 per determinare a quale canale operatore eseguire l'aggiornamento.
2. Opzionale: Se si sta eseguendo l'aggiornamento da una versione di IBM MQ Operator precedente a 1.5 a IBM MQ Operator 1.5 o successiva, è necessario prima aggiornare la versione di IBM Cloud Pak foundational services.

   Per ulteriori informazioni, consultare “Aggiornamento di IBM Cloud Pak foundational services utilizzando la console web Red Hat OpenShift” a pagina 75.
3. Aggiornare IBM MQ Operator. Le nuove versioni principali o secondarie di IBM MQ Operator vengono fornite tramite i nuovi canali di sottoscrizione. Per aggiornare l'operatore a una nuova versione principale o secondaria, sarà necessario aggiornare il canale selezionato nella sottoscrizione IBM MQ Operator .
   a) Dal riquadro di navigazione, fare clic su **Operatori** > **Operatori installati**.

      Vengono visualizzati tutti gli operatori installati nel progetto specificato.
   b) Selezionare **Operatore IBM MQ**
   c) Passare alla scheda **Sottoscrizione**
   d) Fare clic su **Canale**

      Viene visualizzata la finestra **Modifica canale di aggiornamento sottoscrizione** .
   e) Selezionare il canale desiderato e fare clic su **Salva**.

      L'operatore eseguirà l'aggiornamento alla versione più recente disponibile per il nuovo canale. Vedere “Supporto versione per IBM MQ Operator” a pagina 7.

Operazioni successive

Se è stato eseguito l'aggiornamento a IBM Cloud Pak foundational services 3.7, qualsiasi gestore code che utilizza una licenza IBM Cloud Pak for Integration dovrà essere aggiornato o riavviato. Per ulteriori informazioni su come eseguire questa operazione, consultare “Aggiornamento di un gestore code IBM MQ utilizzando la console Red Hat OpenShift web” a pagina 77.

OpenShift CP4I *Aggiornamento di IBM Cloud Pak foundational services utilizzando la console web Red Hat OpenShift*

Se si sta eseguendo l'aggiornamento da una versione di IBM MQ Operator precedente a 1.5 a IBM MQ Operator 1.5 o successiva, è necessario prima aggiornare la versione di IBM Cloud Pak foundational services.

Prima di iniziare

Nota: È necessario completare questa attività solo se si sta eseguendo l'aggiornamento da una versione di IBM MQ Operator precedente a 1.5 a IBM MQ Operator 1.5 o successiva.

CP4I Se si dispone di gestori code che utilizzano una licenza IBM Cloud Pak for Integration , dopo questo aggiornamento, sarà richiesto un riavvio del gestore code per accedere alla console Web e verranno visualizzati anche altri errori di accesso alla console Web. È possibile correggere questi errori eseguendo l'aggiornamento al valore più recente di `.spec.version` per la versione scelta IBM MQ , una volta completato l'aggiornamento dell'operatore.

CP4I Se si dispone di gestori code esistenti e si utilizza IBM Cloud Pak for Integration Operations Dashboard, consultare “Distribuzione o aggiornamento di IBM MQ 9.2.2 o 9.2.3 con l'integrazione di Operations Dashboard in IBM Cloud Pak for Integration 2021.4” a pagina 111 prima dell'aggiornamento.

Procedura

1. Accedi alla tua console web del cluster Red Hat OpenShift .

2. Dal riquadro di navigazione, fare clic su **Operatori** > **Operatori installati**.

   Vengono visualizzati tutti gli operatori installati nel progetto specificato.
3. Selezionare **IBM Cloud Pak foundational services Operatore**. Si noti che prima della versione 3.7, questo era chiamato **IBM Common Services Operator**
4. Passare alla scheda **Sottoscrizione** .
5. Fare clic sul **Canale**.

   Viene visualizzata la finestra **Modifica canale di aggiornamento sottoscrizione** .
6. Selezionare il canale **v3** , quindi fare clic su **Salva**.

   L'operatore IBM Cloud Pak foundational services esegue l'aggiornamento all'ultima versione disponibile per il nuovo canale. Vedere “Supporto versione per IBM MQ Operator” a pagina 7.

Operazioni successive

È ora possibile aggiornare IBM MQ Operator.

OpenShift CP4I Aggiornamento di IBM MQ Operator utilizzando la CLI di Red Hat OpenShift

IBM MQ Operator può essere aggiornato dalla riga di comando.

Prima di iniziare

Accedi al tuo cluster utilizzando **`cloudctl login`** (per IBM Cloud Pak for Integration) o **`oc login`**.

Prima di poter aggiornare il IBM MQ Operator in un ambiente airgap, è necessario eseguire il mirroring delle immagini IBM Cloud Pak for Integration più recenti. Consultare Preparazione per l'aggiornamento del IBM MQ Operator o del gestore code in un ambiente airgap.

Procedura

1. Esaminare “Supporto versione per IBM MQ Operator” a pagina 7 per determinare a quale canale operatore eseguire l'aggiornamento.
2. Opzionale: Se si sta eseguendo l'aggiornamento da una versione di IBM MQ Operator precedente a 1.5 a IBM MQ Operator 1.5 o successiva, è necessario prima aggiornare la versione di IBM Cloud Pak foundational services.

   Per ulteriori informazioni, consultare “Aggiornamento di IBM Cloud Pak foundational services utilizzando la CLI di Red Hat OpenShift” a pagina 77.
3. Aggiornare IBM MQ Operator. Le nuove versioni principali / secondarie di IBM MQ Operator vengono fornite tramite i nuovi canali di sottoscrizione. Per aggiornare il tuo Operatore ad una nuova versione principale / secondaria, dovrai aggiornare il canale selezionato nella tua sottoscrizione IBM MQ Operator .
   a) Assicurarsi che il canale di aggiornamento IBM MQ Operator richiesto sia disponibile.

   ```
   oc get packagemanifest ibm-mq -o=jsonpath='{.status.channels[*].name}'
   ```

   b) Applicare una patch a `Subscription` per passare al canale di aggiornamento desiderato (dove *vX.Y* è il canale di aggiornamento desiderato identificato nel passo precedente.

   ```
   oc patch subscription ibm-mq --patch '{"spec":{"channel":"vX.Y"}}' --type=merge
   ```

Operazioni successive

Se è stato eseguito l'aggiornamento a IBM Cloud Pak foundational services 3.7, qualsiasi gestore code che utilizza una licenza IBM Cloud Pak for Integration dovrà essere aggiornato o riavviato. Per ulteriori informazioni su come eseguire questa operazione, consultare “Aggiornamento di un gestore code IBM MQ utilizzando la CLI Red Hat OpenShift” a pagina 78.

OpenShift CP4I *Aggiornamento di IBM Cloud Pak foundational services utilizzando la CLI di Red Hat OpenShift*

Se si sta eseguendo l'aggiornamento da una versione di IBM MQ Operator precedente a 1.5 a IBM MQ Operator 1.5 o successiva, è necessario prima aggiornare la versione di IBM Cloud Pak foundational services.

Prima di iniziare

Nota: È necessario completare questa attività solo se si sta eseguendo l'aggiornamento da una versione di IBM MQ Operator precedente a 1.5 a IBM MQ Operator 1.5 o successiva.

CP4I Se si dispone di gestori code che utilizzano una licenza IBM Cloud Pak for Integration , dopo questo aggiornamento, verrà richiesto un riavvio del gestore code per accedere alla console Web e verranno visualizzati anche altri errori di accesso alla console Web. È possibile correggere questi errori eseguendo l'aggiornamento al valore più recente di `.spec.version` per la versione scelta IBM MQ , una volta completato l'aggiornamento dell'operatore.

CP4I Se si dispone di gestori code esistenti e si utilizza IBM Cloud Pak for Integration Operations Dashboard, consultare “Distribuzione o aggiornamento di IBM MQ 9.2.2 o 9.2.3 con l'integrazione di Operations Dashboard in IBM Cloud Pak for Integration 2021.4” a pagina 111 prima dell'aggiornamento.

Procedura

1. Accedi al tuo cluster utilizzando **`cloudctl login`** (per IBM Cloud Pak for Integration) o **`oc login`**.
2. Verificare che v3 IBM Cloud Pak foundational services Upgrade Channel sia disponibile.

   ```
   oc get packagemanifest -n ibm-common-services ibm-common-service-operator
   -o=jsonpath='{.status.channels[*].name}'
   ```

3. Correggere `Subscription` per passare al canale di aggiornamento desiderato: `v3`

   ```
   oc patch subscription ibm-common-service-operator --patch '{"spec":{"channel":"v3"}}' --
   type=merge
   ```

Operazioni successive

È ora possibile aggiornare IBM MQ Operator.

OpenShift CP4I Aggiornamento di un gestore code IBM MQ utilizzando la console Red Hat OpenShift web

Un gestore code IBM MQ , distribuito utilizzando IBM MQ Operator, può essere aggiornato in Red Hat OpenShift utilizzando l'hub operatore.

Prima di iniziare

- Accedi alla tua console web del cluster Red Hat OpenShift .
- Assicurarsi che IBM MQ Operator stia utilizzando il canale di aggiornamento desiderato. Vedere “Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 71.

Prima di aggiornare il gestore code in un ambiente airgap, è necessario eseguire il mirroring delle immagini IBM Cloud Pak for Integration più recenti. Consultare Preparazione per l'aggiornamento del IBM MQ Operator o del gestore code in un ambiente airgap.

Procedura

1. Dal riquadro di navigazione, fare clic su **Operatori** > **Operatori installati**.
   Vengono visualizzati tutti gli operatori installati nel progetto specificato.

2. Selezionare **Operatore IBM MQ**.
   Viene visualizzata la finestra **Operatore IBM MQ** .
3. Passare alla scheda **Gestore code** .
   Viene visualizzata la finestra **Dettagli gestore code** .
4. Selezionare il gestore code che si desidera aggiornare.
5. Passare alla scheda YAML.
6. Aggiornare i seguenti campi, se necessario, in modo che corrispondano all'aggiornamento della versione del gestore code IBM MQ desiderato.
   - spec.version
   - spec.license.licence

   Consultare “Supporto versione per IBM MQ Operator” a pagina 7 per un'associazione di canali alle versioni IBM MQ Operator e alle versioni del gestore code IBM MQ .
7. Salvare l'YAML del gestore code aggiornato.

OpenShift CP4I Aggiornamento di un gestore code IBM MQ utilizzando la CLI Red Hat OpenShift

Un gestore code IBM MQ , distribuito utilizzando IBM MQ Operator, può essere aggiornato in Red Hat OpenShift utilizzando la riga comandi.

Prima di iniziare

Per completare queste operazioni, è necessario essere un amministratore del cluster.

- Accedi alla CLI (command line interface) Red Hat OpenShift utilizzando `oc login`.
- Assicurarsi che IBM MQ Operator stia utilizzando il canale di aggiornamento desiderato. Vedere “Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 71.

Prima di aggiornare il gestore code in un ambiente airgap, è necessario eseguire il mirroring delle immagini IBM Cloud Pak for Integration più recenti. Consultare Preparazione per l'aggiornamento del IBM MQ Operator o del gestore code in un ambiente airgap.

Procedura

Modificare la risorsa **QueueManager** per aggiornare i seguenti campi, se necessario, in modo che corrispondano all'aggiornamento della versione del gestore code IBM MQ desiderato.

- spec.version
- spec.license.licence

Consultare “Supporto versione per IBM MQ Operator” a pagina 7 per un'associazione di canali alle versioni IBM MQ Operator e alle versioni del gestore code IBM MQ .

Utilizzare il seguente comando:

```
oc edit queuemanager my_qmgr
```

dove *my_qmgr* è il nome della risorsa QueueManager che si desidera aggiornare.

CP4I Aggiornamento di un gestore code IBM MQ in Red Hat OpenShift utilizzando la piattaforma Navigator

Un gestore code IBM MQ , distribuito utilizzando IBM MQ Operator, può essere aggiornato in Red Hat OpenShift utilizzando IBM Cloud Pak for Integration Platform Navigator.

Prima di iniziare

- Accedere al IBM Cloud Pak for Integration Platform Navigator nello spazio dei nomi che contiene il gestore code che si desidera aggiornare.
- Assicurarsi che IBM MQ Operator stia utilizzando il canale di aggiornamento desiderato. Vedere “Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 71.

Prima di aggiornare il gestore code in un ambiente airgap, è necessario eseguire il mirroring delle immagini IBM Cloud Pak for Integration più recenti. Consultare Preparazione per l'aggiornamento del IBM MQ Operator o del gestore code in un ambiente airgap.

Procedura

1. Dalla home page IBM Cloud Pak for Integration Platform Navigator , fare clic sulla scheda **Runtime** .
2. I gestori code con aggiornamenti disponibili hanno una **i** blu accanto a **Versione**. Fare clic su **i** per visualizzare **Nuova versione disponibile**.
3. Fare clic sui tre punti all'estrema destra del gestore code che si desidera aggiornare, quindi fare clic su **Modifica versione**.
4. In **Seleziona un nuovo canale o versione**, selezionare la versione di aggiornamento richiesta.
5. Fare clic su **Modifica versione**.

Risultati

Il gestore code è aggiornato.

OpenShift CP4I Distribuzione e configurazione dei gestori code utilizzando IBM MQ Operator

IBM MQ 9.1.5 e versioni successive vengono distribuite a Red Hat OpenShift utilizzando IBM MQ Operator.

Informazioni su questa attività

Procedura

- “Preparazione del tuo progetto Red Hat OpenShift per IBM MQ” a pagina 79.
- “Distribuzione di un gestore code su un cluster Red Hat OpenShift Container Platform” a pagina 81.

OpenShift CP4I Preparazione del tuo progetto Red Hat OpenShift per IBM MQ

Prepara il tuo cluster Red Hat OpenShift Container Platform in modo che possa distribuire un gestore code.

Procedura

- “Preparazione del tuo progetto Red Hat OpenShift per IBM MQ utilizzando la console Web Red Hat OpenShift” a pagina 80.
- “Preparazione del tuo progetto di Red Hat OpenShift per IBM MQ utilizzando la CLI Red Hat OpenShift” a pagina 80.

Attività correlate

“Distribuzione di un gestore code su un cluster Red Hat OpenShift Container Platform” a pagina 81
Utilizzare la risorsa personalizzata QueueManager per distribuire un gestore code su un cluster Red Hat OpenShift Container Platform.

OpenShift CP4I *Preparazione del tuo progetto Red Hat OpenShift per IBM MQ utilizzando la console Web Red Hat OpenShift*

Prepara il tuo cluster Red Hat OpenShift Container Platform, in modo che sia pronto a distribuire un gestore code utilizzando IBM MQ Operator. Questa attività deve essere completata da un amministratore del progetto.

Prima di iniziare

Nota: Se si prevede di utilizzare IBM MQ in un progetto con altri componenti IBM Cloud Pak for Integration già installati, potrebbe non essere necessario seguire queste istruzioni.

Accedi alla tua console web del cluster Red Hat OpenShift .

Informazioni su questa attività

Le immagini IBM MQ Operator vengono estratte da un registro del contenitore che esegue un controllo della titolarità della licenza. Questo controllo necessita di una chiave di titolarità memorizzata in un segreto di pull `docker-registry` . Se non si dispone ancora di una chiave di titolarità, seguire queste istruzioni per ottenere una chiave di titolarità e creare un segreto di pull.

Procedura

1. Ottieni la chiave di titolarità assegnata al tuo ID.
   a) Accedere a MyIBM Container Software Library con l'ID e la password IBM associati al software autorizzato.
   b) Nella sezione **Chiavi di titolarità** , selezionare **Copia chiave** per copiare la chiave di titolarità negli appunti.
2. Creare un segreto contenente la chiave di titolarità, nel progetto in cui si desidera distribuire il gestore code.
   a) Dal riquadro di navigazione, fai clic su **Workloads** > **Secret**.
   Viene visualizzata la pagina Segreti.
   b) Nel menu a discesa **Progetto** , selezionare il progetto in cui si desidera installare IBM MQ
   c) Fai clic sul pulsante **Create** e seleziona **Image Pull Secret**
   d) Nel campo **Nome** , immettere `ibm-entitlement-key`
   e) Nel campo **Indirizzo server di registro** , immettere `cp.icr.io`
   f) Nel campo **Nome utente** , immettere `cp`
   g) Nel campo **Password** , immettere la chiave di titolarità copiata nel passo precedente
   h) Nel campo **Email** , immettere l'ID IBM associato al software autorizzato

Operazioni successive

“Distribuzione di un gestore code mediante la console Web Red Hat OpenShift” a pagina 83

OpenShift CP4I *Preparazione del tuo progetto di Red Hat OpenShift per IBM MQ utilizzando la CLI Red Hat OpenShift*

Prepara il tuo cluster Red Hat OpenShift Container Platform, in modo che sia pronto a distribuire un gestore code utilizzando IBM MQ Operator. Questa attività deve essere completata da un amministratore del progetto.

Prima di iniziare

Nota: Se si prevede di utilizzare IBM MQ in un progetto con altri componenti IBM Cloud Pak for Integration già installati, potrebbe non essere necessario seguire queste istruzioni.

Accedi al tuo cluster utilizzando **`cloudctl login`** (per IBM Cloud Pak for Integration) o **`oc login`**.

Informazioni su questa attività

Le immagini IBM MQ Operator vengono estratte da un registro del contenitore che esegue un controllo della titolarità della licenza. Questo controllo necessita di una chiave di titolarità memorizzata in un segreto di pull `docker-registry` . Se non si dispone ancora di una chiave di titolarità, seguire queste istruzioni per ottenere una chiave di titolarità e creare un segreto di pull.

Procedura

1. Ottieni la chiave di titolarità assegnata al tuo ID.
   a) Accedere a MyIBM Container Software Library con l'ID e la password IBM associati al software autorizzato.
   b) Nella sezione **Chiavi di titolarità** , selezionare **Copia chiave** per copiare la chiave di titolarità negli appunti.
2. Creare un segreto contenente la chiave di titolarità, nel progetto in cui si desidera distribuire il gestore code.

   Eseguire il seguente comando, dove *< entitlement-key>* è la chiave richiamata nel passo 1 e *< user-email>* è l'ID IBM associato al software autorizzato.

```
oc create secret docker-registry ibm-entitlement-key \
--docker-server=cp.icr.io \
--docker-username=cp \
--docker-password=<entitlement-key> \
--docker-email=<user-email>
```

Operazioni successive

“Distribuzione di un gestore code utilizzando la CLI Red Hat OpenShift” a pagina 84

OpenShift CP4I Distribuzione di un gestore code su un cluster Red Hat OpenShift Container Platform

Utilizzare la risorsa personalizzata QueueManager per distribuire un gestore code su un cluster Red Hat OpenShift Container Platform.

Procedura

- CP4I
  “Distribuzione di un gestore code utilizzando IBM Cloud Pak for Integration Platform Navigator” a pagina 81.
- OpenShift
  “Distribuzione di un gestore code mediante la console Web Red Hat OpenShift” a pagina 83.
- OpenShift
  “Distribuzione di un gestore code utilizzando la CLI Red Hat OpenShift” a pagina 84.

Attività correlate

“Esempi per configurare un gestore code” a pagina 85
È possibile configurare un gestore code modificando il contenuto della risorsa personalizzata QueueManager .

CP4I *Distribuzione di un gestore code utilizzando IBM Cloud Pak for Integration Platform Navigator*

Utilizzare la risorsa personalizzata QueueManager per distribuire un gestore code su un cluster Red Hat OpenShift Container Platform utilizzando IBM Cloud Pak for Integration Platform Navigator. Questa attività deve essere completata da un amministratore del progetto

Prima di iniziare

In un browser, avviare IBM Cloud Pak for Integration Platform Navigator.

Se questa è la prima distribuzione di un gestore code in questo progetto Red Hat OpenShift , seguire la procedura per “Preparazione del tuo progetto Red Hat OpenShift per IBM MQ” a pagina 79.

Procedura

1. Distribuire un gestore code.

   Il seguente esempio distribuisce un gestore code "avvio rapido", che utilizza memoria effimera (non persistente) e disattiva la sicurezza di MQ . I messaggi non saranno resi persistenti durante i riavvii del gestore code. È possibile modificare la configurazione per modificare molte impostazioni del gestore code.

   a) In IBM Cloud Pak for Integration Platform Navigator, fare clic su **Amministrazione** , quindi su **Runtime di integrazione**. Nelle versioni precedenti di IBM Cloud Pak for Integration Platform Navigator, fare clic su **Runtime and instances**.

   b) Fai clic su **Crea istanza**.

   c) Selezionare **Messaggistica**e fare clic su **Avanti**. Nelle versioni precedenti di IBM Cloud Pak for Integration Platform Navigator, fare clic su **Gestore code**e fare clic su **Avanti**.

   Viene visualizzato il modulo per creare un'istanza di un `QueueManager` .

   Nota: È anche possibile fare clic su **Codice** per visualizzare o modificare l'YAML di configurazione `QueueManager` .

   d) Nella sezione **Dettagli** , controllare o aggiornare il campo **Nome** e specificare lo **Spazio dei nomi** in cui creare l'istanza del gestore code.

   e) Se si accetta l'accordo di licenza IBM Cloud Pak for Integration , modificare **Accettazione licenza** in **On**.

   È necessario accettare la licenza per distribuire un gestore code.

   f) Nella sezione **Gestore code** , verificare o aggiornare il **Nome** del gestore code sottostante. Nelle versioni precedenti di IBM Cloud Pak for Integration Platform Navigator, utilizzare la sezione **Configurazione gestore code** .

   Per impostazione predefinita, il nome del gestore code utilizzato dalle applicazioni client IBM MQ sarà uguale al nome di `QueueManager`, ma con tutti i caratteri non validi (come i trattini) rimossi.

   g) Fai clic su **Crea**

   Viene ora visualizzato l'elenco dei gestori code nel progetto corrente (spazio dei nomi). Il nuovo `QueueManager` deve avere lo stato `Pending`

2. Verificare che il gestore code sia in esecuzione

   La creazione è completa quando lo stato di `QueueManager` è `Running`.

Attività correlate

“Configurazione di un instradamento per la connessione a un gestore code dall'esterno di un cluster Red Hat OpenShift” a pagina 108
Hai bisogno di un instradamento Red Hat OpenShift per connettere un'applicazione a un gestore code IBM MQ dall'esterno di un cluster Red Hat OpenShift . È necessario abilitare TLS sul gestore code e sull'applicazione client IBM MQ , perché SNI è disponibile solo nel protocollo TLS quando viene utilizzato un protocollo TLS 1.2 o superiore. Red Hat OpenShift Container Platform Router utilizza SNI per instradare le richieste al gestore code IBM MQ .

“Connessione al IBM MQ Console distribuito in un cluster Red Hat OpenShift” a pagina 115
Modalità di connessione a IBM MQ Console di un gestore code distribuito su un cluster Red Hat OpenShift Container Platform .

OpenShift CP4I *Distribuzione di un gestore code mediante la console Web Red Hat OpenShift*

Utilizzare la risorsa personalizzata QueueManager per distribuire un gestore code su un cluster Red Hat OpenShift Container Platform utilizzando la console web Red Hat OpenShift . Questa attività deve essere completata da un amministratore del progetto

Prima di iniziare

Accedi alla tua console web del cluster Red Hat OpenShift . Sarà necessario selezionare un progetto esistente (namespace) da utilizzare o crearne uno nuovo.

Se questa è la prima distribuzione di un gestore code in questo progetto Red Hat OpenShift , seguire la procedura per “Preparazione del tuo progetto Red Hat OpenShift per IBM MQ” a pagina 79.

Procedura

1. Distribuire un gestore code.

   Il seguente esempio distribuisce un gestore code "avvio rapido", che utilizza memoria effimera (non persistente) e disattiva la sicurezza di MQ . I messaggi non saranno resi persistenti durante i riavvii del gestore code. È possibile modificare la configurazione per modificare molte impostazioni del gestore code.

   a) Nella console web Red Hat OpenShift , dal pannello di navigazione, fare clic su **Operatori** > **Operatori installati**

   b) Fare clic su **IBM MQ**.

   c) Fare clic sulla scheda **Gestore code** .

   d) Fare clic sul pulsante **Crea QueueManager** .

   Viene visualizzato un editor YAML, contenente YAML di esempio per una risorsa `QueueManager` .

   Nota: È anche possibile fare clic su **Modifica modulo** per visualizzare o modificare la configurazione di `QueueManager` .

   e) Se si accetta l'accordo di licenza, modificare **Accettazione licenza** in **On**.

   IBM MQ è disponibile con diverse licenze. Per ulteriori informazioni sulle licenze valide, consultare “Riferimento per la licenza per mq.ibm.com/v1beta1” a pagina 126. È necessario accettare la licenza per distribuire un gestore code.

   f) Fai clic su **Crea**

   Viene ora visualizzato l'elenco dei gestori code nel progetto corrente (spazio dei nomi). Il nuovo `QueueManager` deve essere in uno stato `Pending` .

2. Verificare che il gestore code sia in esecuzione

   La creazione è completa quando lo stato di `QueueManager` è `Running`.

Attività correlate

“Configurazione di un instradamento per la connessione a un gestore code dall'esterno di un cluster Red Hat OpenShift” a pagina 108
Hai bisogno di un instradamento Red Hat OpenShift per connettere un'applicazione a un gestore code IBM MQ dall'esterno di un cluster Red Hat OpenShift . È necessario abilitare TLS sul gestore code e sull'applicazione client IBM MQ , perché SNI è disponibile solo nel protocollo TLS quando viene utilizzato un protocollo TLS 1.2 o superiore. Red Hat OpenShift Container Platform Router utilizza SNI per instradare le richieste al gestore code IBM MQ .

“Connessione al IBM MQ Console distribuito in un cluster Red Hat OpenShift” a pagina 115
Modalità di connessione a IBM MQ Console di un gestore code distribuito su un cluster Red Hat OpenShift Container Platform .

OpenShift CP4I *Distribuzione di un gestore code utilizzando la CLI Red Hat OpenShift*

Utilizza la risorsa di personalizzazione QueueManager per distribuire un gestore code su un cluster Red Hat OpenShift Container Platform utilizzando la CLI (command line interface). Questa attività deve essere completata da un amministratore del progetto

Prima di iniziare

È necessario installare la CLI (command - line interface) Red Hat OpenShift Container Platform.

Accedi al tuo cluster utilizzando **`cloudctl login`** (per IBM Cloud Pak for Integration) o **`oc login`**.

Se questa è la prima distribuzione di un gestore code in questo progetto Red Hat OpenShift , seguire la procedura per “Preparazione del tuo progetto Red Hat OpenShift per IBM MQ” a pagina 79.

Procedura

1. Distribuire un gestore code.

   Il seguente esempio distribuisce un gestore code "avvio rapido", che utilizza memoria effimera (non persistente) e disattiva la sicurezza di MQ . I messaggi non saranno resi persistenti durante i riavvii del gestore code. È possibile modificare il contenuto di YAML per modificare molte impostazioni del gestore code.

   a) Crea un file YAML di `QueueManager`

   Ad esempio, per installare un gestore code di base in IBM Cloud Pak for Integration, creare un file "mq-quickstart.yaml" con i seguenti contenuti:

```
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"
```

   Importante: Se si accetta l'accordo di licenza di IBM Cloud Pak for Integration , modificare `accept: false` in `accept: true`. Consultare “Riferimento per la licenza per mq.ibm.com/v1beta1” a pagina 126 per i dettagli sulla licenza.

   Questo esempio include anche un server Web distribuito con il gestore code, con la console web abilitata con Single Sign - On con IBM Cloud Pak Identity and Access Manager.

   Per installare un gestore code di base indipendentemente da IBM Cloud Pak for Integration, crea il filemq-quickstart.yaml" con il seguente contenuto:

```
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"
```

Importante:se si accetta l'accordo di licenza di MQ , modificare `accept: false` in `accept: true`. Consultare “Riferimento per la licenza per mq.ibm.com/v1beta1” a pagina 126 per i dettagli sulla licenza.

b) Creare l'oggetto `QueueManager`

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

2. Verificare che il gestore code sia in esecuzione

   È possibile convalidare la distribuzione eseguendo

   ```
   oc describe queuemanager <QueueManagerResourceName>
   ```

   , e quindi controllare lo stato.

   Ad esempio, eseguire

   ```
   oc describe queuemanager quickstart
   ```

   , e verificare che il campo `status.Phase` indichi `Running`

Attività correlate

“Configurazione di un instradamento per la connessione a un gestore code dall'esterno di un cluster Red Hat OpenShift” a pagina 108
Hai bisogno di un instradamento Red Hat OpenShift per connettere un'applicazione a un gestore code IBM MQ dall'esterno di un cluster Red Hat OpenShift . È necessario abilitare TLS sul gestore code e sull'applicazione client IBM MQ , perché SNI è disponibile solo nel protocollo TLS quando viene utilizzato un protocollo TLS 1.2 o superiore. Red Hat OpenShift Container Platform Router utilizza SNI per instradare le richieste al gestore code IBM MQ .

“Connessione al IBM MQ Console distribuito in un cluster Red Hat OpenShift” a pagina 115
Modalità di connessione a IBM MQ Console di un gestore code distribuito su un cluster Red Hat OpenShift Container Platform .

OpenShift CP4I Esempi per configurare un gestore code

È possibile configurare un gestore code modificando il contenuto della risorsa personalizzata QueueManager .

Informazioni su questa attività

Utilizzare i seguenti esempi per configurare un gestore code utilizzando il file YAML QueueManager .

Procedura

- “Esempio: fornitura di file MQSC e INI” a pagina 86
- “Esempio: configurazione di TLS” a pagina 87

OpenShift CP4I *Esempio: fornitura di file MQSC e INI*

Questo esempio crea una Kubernetes ConfigMap ConfigMap contenente due file MQSC e un file INI. Viene quindi distribuito un gestore code che elabora questi file MQSC e INI.

Informazioni su questa attività

I file MQSC e INI possono essere forniti quando un gestore code viene distribuito. I dati MQSC e INI devono essere definiti in una o più Kubernetes ConfigMaps e Segreti. Questi devono essere creati nello spazio dei nomi (progetto) in cui verrà distribuito il gestore code.

Nota: Un segreto Kubernetes deve essere utilizzato quando il file MQSC o INI contiene dati sensibili.

Fornire MQSC e INI in questo modo richiede IBM MQ Operator 1.1 o superiore.

Esempio

Il seguente esempio crea una Kubernetes ConfigMap che contiene due file MQSC e un file INI. Viene quindi distribuito un gestore code che elabora questi file MQSC e INI.

Esempio ConfigMap - applica il seguente YAML nel tuo 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
```

Esempio QueueManager - distribuisci il tuo gestore code con la seguente configurazione, utilizzando la riga di comando o 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
```

Importante: Se si accetta l'accordo di licenza di IBM Cloud Pak for Integration , modificare `accept: false` in `accept: true`. Per dettagli sulla licenza, consultare Riferimento alle licenze per mq.ibm.com/v1beta1 .

Informazioni aggiuntive:

- Un gestore code può essere configurato per utilizzare un singolo Kubernetes ConfigMap o segreto (come mostrato in questo esempio) o più Kubernetes ConfigMaps e segreti.
- È possibile scegliere di utilizzare tutti i dati MQSC e INI da una Kubernetes ConfigMap o da un segreto (come mostrato in questo esempio) oppure configurare ciascun gestore code per utilizzare solo un sottoinsieme dei file disponibili.
- I file MQSC e INI vengono elaborati in ordine alfabetico in base alla loro chiave. Quindi `example1.mqsc` verrà sempre elaborato prima di `example2.mqsc`, indipendentemente dall'ordine in cui vengono visualizzati nella configurazione del gestore code.
- Se più file MQSC o INI hanno la stessa chiave, su più Kubernetes ConfigMaps o Secrets, questa serie di file viene elaborata in base all'ordine in cui i file sono definiti nella configurazione del gestore code.

OpenShift CP4I Linux *Esempio: configurazione di TLS*

Questo esempio distribuisce un gestore code in Red Hat OpenShift Container Platform utilizzando IBM MQ Operator. La comunicazione TLS unidirezionale è configurata tra un client di esempio e il gestore code. L'esempio illustra la corretta configurazione inserendo e ottenendo messaggi.

Prima di iniziare

Per completare questo esempio, è necessario prima aver completato i prerequisiti riportati di seguito:

- Installa IBM MQ cliente aggiungi `samp/bin` e `bin` al *PATH*. Sono necessarie le applicazioni **`runmqakm`**, **`amqsputc`** e **`amqsgetc`** , che possono essere installate come parte di IBM MQ client come segue:
  - Linux Windows Per Windows e Linux: installare il client ridistribuibile IBM MQ per il proprio sistema operativo da https://ibm.biz/mq92redistclients
  - macOS Per Mac: scaricare e configurare IBM MQ MacOS Toolkit: https://developer.ibm.com/tutorials/mq-macos-dev/
- Installare lo strumento OpenSSL per il proprio sistema operativo.
- Creare un progetto / spazio dei nomi Red Hat OpenShift Container Platform (OCP) per questo esempio.
- Sulla riga comandi, accedere al cluster OCP e passare allo spazio dei nomi precedente.
- Assicurarsi che il file IBM MQ Operator sia installato e disponibile nello spazio dei nomi precedente.

Informazioni su questa attività

Questo esempio fornisce una risorsa personalizzata YAML che definisce un gestore code da distribuire in Red Hat OpenShift Container Platform. Descrive inoltre i passi aggiuntivi richiesti per distribuire il gestore code con TLS abilitato. Al completamento, l'inserimento e il richiamo dei messaggi convalida che il gestore code sia configurato con TLS.

Creare una chiave privata TLS e i certificati per il server IBM MQ

I seguenti esempi di codice mostrano come creare un certificato autofirmato per il gestore code e come aggiungere il certificato a un database di chiavi per agire come truststore per il client. Se si dispone già di una chiave privata e di un certificato, è possibile utilizzarli.

Si noti che i certificati autofirmati devono essere utilizzati solo per scopi di sviluppo.

Crea una chiave privata autofirmata e un certificato pubblico nella directory corrente

Esegui il seguente comando:

```
openssl req -newkey rsa:2048 -nodes -keyout tls.key -subj "/CN=localhost" -x509 -days 3650
-out tls.crt
```

Aggiungere la chiave pubblica del server a un database di chiavi client

Il database di chiavi viene utilizzato come truststore per l'applicazione client.

Creare il database delle chiavi client:

```
runmqakm -keydb -create -db clientkey.kdb -pw password -type cms -stash
```

Aggiungere la chiave pubblica generata in precedenza al database delle chiavi client:

```
runmqakm -cert -add -db clientkey.kdb -label mqservercert -file tls.crt -format ascii
-stashed
```

Configura certificati TLS per la distribuzione del gestore code

In questo modo, il tuo gestore code può fare riferimento e applicare la chiave e il certificato, creare un segreto TLS Kubernetes , facendo riferimento ai file precedentemente creati. Quando si esegue questa operazione, assicurarsi di essere nello spazio dei nomi creato prima di iniziare questa attività.

```
oc create secret tls example-tls-secret --key="tls.key" --cert="tls.crt"
```

Crea una mappa di configurazione contenente i comandi MQSC

Crea una mappa di configurazione Kubernetes contenente i comandi MQSC per la creazione di una nuova coda e di un canale SVRCONN e per aggiungere un record di autenticazione di canale che consenta l'accesso al canale bloccando solo gli utenti denominati *nobody*.

Notare che questo approccio deve essere utilizzato solo per scopi di sviluppo.

Assicurati di essere nello spazio dei nomi che hai creato in precedenza (vedi Prima di iniziare), quindi immetti il seguente YAML nell'IU OCP o utilizzando la riga di comando.

```
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)
```

Crea l'instradamento OCP richiesto

Assicurarsi di essere nello spazio dei nomi creato prima di iniziare questa attività, quindi immettere il seguente YAML nell'interfaccia utente OCP o utilizzando la riga comandi.

```
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
```

Si noti che Red Hat OpenShift Container Platform Router utilizza SNI per instradare le richieste al gestore code IBM MQ . Se si modifica il nome del canale specificato in MQSC nella mappa di configurazione creata in precedenza, è necessario modificare anche il campo host qui e nel file CCDT creato in seguito. Per ulteriori informazioni, consultare “Configurazione di un instradamento per la connessione a un gestore code dall'esterno di un cluster Red Hat OpenShift” a pagina 108.

Distribuisci il gestore code

Importante: in questo esempio utilizziamo la variabile *MQSNOAUT* per disabilitare l'autorizzazione sul gestore code, che ci consente di concentrarci sui passi richiesti per connettere un client utilizzando TLS. Ciò non è consigliato in una distribuzione di produzione di IBM MQ, perché fa sì che tutte le applicazioni che si collegano abbiano pieni poteri di gestione, senza alcun meccanismo per ridurre le autorizzazioni per le singole applicazioni.

Creare un nuovo gestore code utilizzando la seguente risorsa personalizzata YAML. Tenere presente che fa riferimento alla mappa di configurazione e al segreto creati precedentemente, nonché alla variabile *MQSNOAUT* .

Assicurati di essere nello spazio dei nomi creato prima di iniziare questa attività, quindi immetti il seguente YAML nella IU OCP, utilizzando la riga di comando o utilizzando IBM Cloud Pak for Integration Platform Navigator. Verificare che sia stata specificata la licenza corretta e accettarla modificando `false` in `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
```

Confermare che il gestore code è in esecuzione

Il gestore code è in fase di distribuzione. Confermare che si trova nello stato `Running` prima di procedere. Ad esempio:

```
oc get qmgr secureqm
```

Verifica la connessione al gestore code

Per confermare che il gestore code è configurato per la comunicazione TLS unidirezionale, utilizzare le applicazioni di esempio **amqsputc** e **amqsgetc** :

Trova il nome host del gestore code

Utilizzare il seguente comando per trovare il nome host completo del gestore code per l'instradamento `secureqm-ibm-mq-qm`:

```
oc get routes secureqm-ibm-mq-qm
```

Specificare i dettagli del gestore code

Creare un file `CCDT.JSON` che specifichi i dettagli del gestore code. Sostituire il valore host con il nome host del passo precedente.

```
{
    "channel":
    [
        {
            "name": "SECUREQMCHL",
            "clientConnection":
```

```
            {
                "connection":
                [
                    {
                        "host": "<hostname from previous step>",
                        "port": 443
                    }
                ],
                "queueManager": "SECUREQM"
            },
            "transmissionSecurity":
            {
              "cipherSpecification": "ECDHE_RSA_AES_128_CBC_SHA256"
            },
            "type": "clientConnection"
        }
    ]
}
```

Esporta variabili di ambiente

Esportare le seguenti variabili di ambiente, nel modo appropriato per il proprio sistema operativo. Queste variabili verranno lette da **amqsputc** e **amqsgetc**.

Aggiornare il percorso dei file sul sistema:

```
export MQCCDTURL='<full path to file>/CCDT.JSON'
export MQSSLKEYR='<full path to file>/clientkey'
```

Inserire i messaggi nella coda

Esegui il seguente comando:

```
amqsputc EXAMPLE.QUEUE SECUREQM
```

Se la connessione al gestore code ha esito positivo, viene emessa la seguente risposta:

```
target queue is EXAMPLE.QUEUE
```

Inserire diversi messaggi nella coda, immettendo del testo e premendo ogni volta **Invio** .

Per terminare, premere due volte **Invio** .

Richiamare i messaggi dalla coda

Esegui il seguente comando:

```
amqsgetc EXAMPLE.QUEUE SECUREQM
```

I messaggi aggiunti nel passo precedente sono stati utilizzati e vengono emessi.

Dopo pochi secondi, il comando esce.

Congratulazioni, hai distribuito correttamente un gestore code con TLS abilitato e hai mostrato che puoi inserire e richiamare in modo sicuro i messaggi al gestore code da un client.

OpenShift CP4I *Esempio: personalizzazione delle annotazioni del servizio di licenza*

IBM MQ Operator aggiunge automaticamente le annotazioni IBM License Service alle risorse distribuite. Questi sono monitorati da IBM License Servicee vengono generati report che corrispondono alla titolarità richiesta.

Informazioni su questa attività

Le annotazioni aggiunte da IBM MQ Operator sono quelle previste in situazioni standard e si basano sui valori della licenza selezionati durante la distribuzione di un gestore code.

Esempio

Se **License** è impostata su `L-RJON-BZFQU2 (IBM Cloud Pak for Integration 2021.2.1)`e **Use** è impostato su `NonProduction`, vengono applicate le seguenti annotazioni:

- cloudpakId: c8b82d189e7545f0892db9ef2731b90d
- cloudpakName: IBM Cloud Pak for Integration
- Contenitori productCharged: qmgr
- productCloudpakRapporto: '4:1'
- productID: 21dfe9a0f00f444f888756d835334909
- productName: IBM MQ Advanced per Non - Production
- productMetric: VIRTUAL_PROCESSOR_CORE
- productVersion: 9.2.3.0

All'interno di IBM Cloud Pak for Integration, le distribuzione di IBM App Connect Enterprise includono una titolarità limitata per IBM MQ. In queste situazioni, queste annotazioni devono essere sovrascritte per garantire che IBM License Service acquisisca l'uso corretto. A tale scopo, utilizzare l'approccio descritto in “Aggiunta di annotazioni ed etichette personalizzate alle risorse del gestore code” a pagina 113.

Ad esempio, se IBM MQ viene distribuito in base alla titolarità IBM App Connect Enterprise , utilizzare l'approccio mostrato nel seguente frammento di codice:

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: mq4ace
  namespace: cp4i
spec:
  annotations:
    productMetric: FREE
```

Ci sono altri due motivi comuni per cui le annotazioni di licenza potrebbero richiedere modifiche:

1. IBM MQ Advanced è incluso nella titolarità di un altro prodotto IBM .
   - In questa situazione, utilizzare l'approccio precedentemente descritto per IBM App Connect Enterprise.
2. IBM MQ viene distribuito con una licenza IBM Cloud Pak for Integration .
   - Se si dispone di una licenza IBM Cloud Pak for Integration , è possibile decidere di distribuire un gestore code nel rapporto IBM MQ o IBM MQ Advanced . Se esegui la distribuzione in un rapporto IBM MQ , devi assicurarti di non utilizzare alcuna funzionalità avanzata come la HA nativa o Advanced Message Security.
   - In questa situazione, utilizzare le seguenti annotazioni per uso di produzione:

     ```
     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
     ```

   - Utilizzare le seguenti annotazioni per uso non di produzione:

     ```
     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 Configurazione dell'alta disponibilità per i gestori code utilizzando IBM MQ Operator

Informazioni su questa attività

Procedura

- V 9.2.3 “HA nativa” a pagina 92.
- V 9.2.3 “Esempio: configurazione di un gestore code HA nativo” a pagina 94.
- “Esempio: configurazione di un gestore code a più istanze” a pagina 102.

CP4I CD V 9.2.3 *HA nativa*

La HA nativa è una soluzione di alta disponibilità nativa (integrata) per IBM MQ adatta per l'utilizzo con l'archiviazione blocchi cloud.

Una configurazione della HA nativa fornisce un gestore code ad alta disponibilità in cui i dati MQ recuperabili (ad esempio, i messaggi) vengono replicati su più serie di memoria, impedendo la perdita a causa di errori di memoria. Il gestore code è costituito da più istanze in esecuzione, una è il leader, le altre sono pronte a subentrare rapidamente in caso di errore, massimizzando l'accesso al gestore code e ai relativi messaggi.

Una configurazione della HA nativa è composta da tre pod Kubernetes , ciascuno con un'istanza del gestore code. Un'istanza è il gestore code attivo, che elabora i messaggi e scrive nel log di ripristino. Ogni volta che viene scritto il log di ripristino, il gestore code attivo invia i dati alle altre due istanze, note come repliche. Ogni replica scrive nel proprio log di ripristino, riconosce i dati e quindi aggiorna i propri dati della coda dal log di ripristino replicato. Se il pod su cui è in esecuzione il gestore code attivo ha esito negativo, una delle istanze di replica del gestore code assume il ruolo attivo e dispone dei dati correnti con cui operare.

Il tipo di log è noto come 'log replicato '. Un log replicato è essenzialmente un log lineare, con la gestione automatica dei log e le immagini di supporto automatiche abilitate. Consultare Tipi di registrazione. Per gestire il log replicato si utilizzano le stesse tecniche utilizzate per la gestione di un log lineare.

Un Kubernetes Service viene utilizzato per instradare connessioni client TCP/IP all'istanza attiva corrente, che è identificata come l'unico pod pronto per il traffico di rete. Ciò si verifica senza che l'applicazione client sia a conoscenza delle diverse istanze.

Tre baccelli sono utilizzati per ridurre notevolmente la possibilità che si verifichi una situazione di divisione del cervello. In un sistema ad alta disponibilità a due pod, lo split - brain può verificarsi quando la connettività tra i due pod si rompe. Senza connettività, entrambi i pod possono eseguire il gestore code contemporaneamente, accumulando dati diversi. Quando la connessione viene ripristinata, ci sarebbero due versioni differenti dei dati (un 'split-brain ') e l'intervento manuale è richiesto per decidere quale serie di dati conservare e quale scartare.

La HA nativa utilizza un sistema a tre pod con quorum per evitare la situazione di split - brain. I pod che possono comunicare con almeno uno degli altri pod formano un quorum. Un gestore code può diventare solo l'istanza attiva su un pod che ha quorum. Il gestore code non può diventare attivo su un pool che non è connesso ad almeno un altro pod, quindi non possono mai esserci due istanze attive contemporaneamente:

- Se un singolo pod ha esito negativo, il gestore code su uno degli altri due pod può eseguire il controllo. Se due pod hanno esito negativo, il gestore code non può diventare l'istanza attiva sul pod rimanente perché il pod non ha quorum (il pod rimanente non può indicare se gli altri due pod hanno avuto esito negativo o se sono ancora in esecuzione e hanno perso la connettività).
- Se un singolo pod perde la connettività, il gestore code non può diventare attivo su questo pod perché il pod non ha quorum. Il gestore code su uno dei due pod rimanenti può assumere il controllo, che

hanno quorum. Se tutti i pod perdono la connettività, il gestore code non è in grado di diventare attivo su nessuno dei pod, perché nessuno dei pod ha il quorum.

Se un pod attivo ha esito negativo e successivamente viene ripristinato, può unirsi nuovamente al gruppo in un ruolo di replica.

La seguente figura mostra una tipica distribuzione con tre istanze di un gestore code distribuite in tre contenitori.

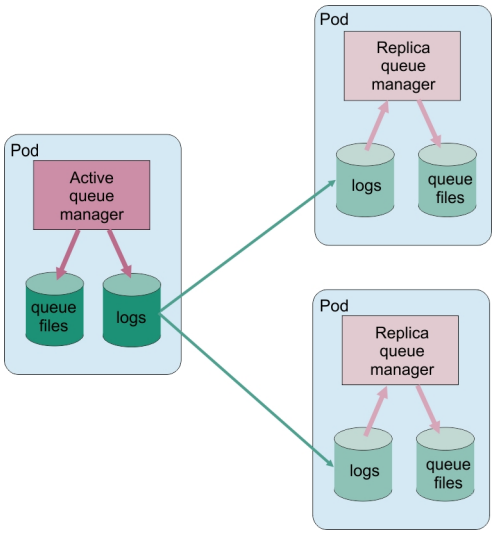


Figura 1. Esempio di configurazione della HA nativa

CP4I CD V9.2.3 *Configurazione della HA nativa utilizzando IBM MQ Operator*

La HA nativa è configurata utilizzando l'API `QueueManager` e le opzioni avanzate sono disponibili utilizzando un file INI.

La HA nativa è configurata utilizzando il `.spec.queueManager.availability` dell'API di `QueueManager` , ad esempio:

```
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
```

il campo `.spec.queueManager.availability.type` deve essere impostato su `NativeHA`.

La HA nativa è disponibile in IBM MQ 9.2.3 o superiore.

In `.spec.queueManager.availability`, è anche possibile configurare un segreto TLS e le cifrature da utilizzare tra le istanze del gestore code durante la replica. Ciò è fortemente consigliato e una guida dettagliata è disponibile in “Esempio: configurazione di un gestore code HA nativo” a pagina 94.

Riferimenti correlati

“Esempio: configurazione di un gestore code HA nativo” a pagina 94
Questo esempio mostra come distribuire un gestore code utilizzando la funzione alta disponibilità nativa in Red Hat OpenShift Container Platform (OCP) utilizzando IBM MQ Operator.

Esempio: configurazione di un gestore code HA nativo

Questo esempio mostra come distribuire un gestore code utilizzando la funzione alta disponibilità nativa in Red Hat OpenShift Container Platform (OCP) utilizzando IBM MQ Operator.

Prima di iniziare

Per completare questo esempio, è necessario prima aver completato i prerequisiti riportati di seguito:

- Installare IBM MQ cliente aggiungere le directory `samp/bin` e `bin` installate al *PATH*. Il client fornisce le applicazioni **`runmqakm`**, **`amqsputc`** e **`amqsgetc`** richieste da questo esempio. Installare IBM MQ client nel modo seguente:
  - 

Per Windows e Linux: installare il client ridistribuibile IBM MQ per il proprio sistema operativo da https://ibm.biz/mq92redistclients
  - 

Per Mac: scaricare e configurare IBM MQ MacOS Toolkit. Consultarehttps://ibm.biz/mqdevmacclient.
- Installare lo strumento OpenSSL per il proprio sistema operativo. È necessario per generare un certificato autofirmato per il gestore code, se non si dispone già di una chiave privata e di un certificato.
- Creare un progetto / spazio dei nomi Red Hat OpenShift Container Platform (OCP) per questo esempio e seguire i passi nell'attività “Preparazione del tuo progetto Red Hat OpenShift per IBM MQ” a pagina 79
- Sulla riga comandi, accedere al cluster OCP e passare allo spazio dei nomi appena creato.
- Accertarsi che IBM MQ Operator sia installato e disponibile nel namespace.
- Configurare una classe di memoria predefinita in OCP, che deve essere utilizzata dal proprio gestore code. Se vuoi completare questa esercitazione senza impostare una classe di archiviazione predefinita, vedi Nota 2: Utilizzo di una classe di archiviazione non predefinita.

Informazioni su questa attività

I gestori code della HA nativa coinvolgono due pod Kubernetes di replica e uno attivo. Questi vengono eseguiti come parte di una serie con stato Kubernetes con esattamente tre repliche e una serie di volumi persistenti Kubernetes . Per ulteriori informazioni sui gestori code della HA nativa, consultare “Alta disponibilità per IBM MQ nei contenitori” a pagina 16.

L'esempio fornisce una risorsa personalizzata YAML che definisce un gestore code HA nativo che utilizza l'archiviazione persistente ed è configurato con TLS. Dopo aver distribuito il gestore code in OCP, si simula l'errore del pod del gestore code attivo. Si vede che si verifica un ripristino automatico e si prova che ha avuto esito positivo inserendo e ricevendo messaggi dopo l'errore.

Esempio

Creare una chiave privata TLS e i certificati per il server MQ

È possibile creare un certificato autofirmato per il gestore code e aggiungere il certificato a un database di chiavi per agire come truststore per il client. Se si dispone già di una chiave privata e di un certificato, è possibile utilizzarli. Notare che è necessario utilizzare i certificati autofirmati solo per scopi di sviluppo.

Per creare una chiave privata autofirmata e un certificato pubblico nella directory corrente, esegui questo comando:

```
openssl req -newkey rsa:2048 -nodes -keyout tls.key -subj "/CN=localhost" -x509 -days 3650
-out tls.crt
```

Crea una chiave privata TLS e i certificati per l'utilizzo interno da parte della HA nativa

I tre pod in un gestore code HA nativo replicano i dati sulla rete. È possibile creare un certificato autofirmato da utilizzare durante la replica interna. Notare che è necessario utilizzare i certificati autofirmati solo per scopi di sviluppo.

Per creare una chiave privata autofirmata e un certificato pubblico nella directory corrente, esegui questo comando:

```
openssl req -newkey rsa:2048 -nodes -keyout nativeha.key -subj "/CN=localhost" -x509 -days
3650 -out nativeha.crt
```

Aggiungi la chiave pubblica del gestore code a un database di chiavi del client

Un database di chiavi client viene utilizzato come truststore per l'applicazione client.

Creare il database delle chiavi client:

```
runmqakm -keydb -create -db clientkey.kdb -pw password -type cms -stash
```

Aggiungere la chiave pubblica generata in precedenza al database delle chiavi client:

```
runmqakm -cert -add -db clientkey.kdb -label mqservercert -file tls.crt -format ascii
-stashed
```

Crea un segreto contenente i certificati TLS per la distribuzione del gestore code

In modo che il tuo gestore code possa fare riferimento e applicare la chiave e il certificato, creare un segreto TLS Kubernetes , facendo riferimento ai file creati in precedenza. Quando si esegue questa operazione, assicurarsi di essere nello spazio dei nomi creato prima di iniziare questa attività.

```
oc create secret tls example-ha-secret --key="tls.key" --cert="tls.crt"
```

Crea un segreto contenente il certificato e la chiave TLS della HA nativa interna

In modo che il tuo gestore code possa fare riferimento e applicare la chiave e il certificato, creare un segreto TLS Kubernetes , facendo riferimento ai file creati in precedenza. Quando si esegue questa operazione, assicurarsi di essere nello spazio dei nomi creato prima di iniziare questa attività.

```
oc create secret tls example-ha-secret-internal --key="nativeha.key" --cert="nativeha.crt"
```

Crea una mappa di configurazione contenente i comandi MQSC

Crea una mappa di configurazione Kubernetes contenente i comandi MQSC per creare una nuova coda e un canale SVRCONN e per aggiungere un record di autenticazione di canale che consenta l'accesso al canale bloccando solo gli utenti denominati *nobody*.

Notare che questo approccio deve essere utilizzato solo per scopi di sviluppo.

Assicurarsi di essere nello spazio dei nomi creato in precedenza (consultare “Prima di iniziare” a pagina 94), quindi immettere il seguente YAML nell'interfaccia utente OCP o utilizzando la riga comandi:

```
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)
```

Configura instradamento

Se si sta utilizzando un IBM MQ client o un toolkit all'indirizzo IBM MQ 9.2.1 o successivo, è possibile configurare l'instradamento al gestore code utilizzando un file di configurazione del gestore code (un file INI). All'interno del file, impostare la variabile *OutboundSNI* per l'instradamento in base al nome host anziché al nome del canale.

Creare un file nella directory in cui si stanno eseguendo i comandi, denominato `mqclient.ini`, contenente esattamente il testo seguente:

```
SSL:
   OutboundSNI=HOSTNAME
```

Non modificare alcun valore in questo file INI. Ad esempio, la stringa HOSTNAME non deve essere modificata.

Per ulteriori dettagli, consultare Stanza SSL del file di configurazione client.

Se si utilizza un IBM MQ client o un toolkit precedente a IBM MQ 9.2.1, è necessario creare un instradamento OCP invece del file di configurazione precedente. Segui la procedura in Nota 1: creazione di un instradamento.

Distribuisci il gestore code

Importante: in questo esempio utilizziamo la variabile *MQSNOAUT* per disabilitare l'autorizzazione sul gestore code, che ci consente di concentrarci sui passi richiesti per connettere un client utilizzando TLS. Ciò non è consigliato in una distribuzione di produzione di IBM MQ, perché fa sì che tutte le applicazioni che si collegano abbiano pieni poteri di gestione, senza alcun meccanismo per ridurre le autorizzazioni per le singole applicazioni.

Copiare e aggiornare il seguente YAML.

- Assicurarsi che sia stata specificata la licenza corretta. Consultare Riferimento alle licenze per mq.ibm.com/v1beta1. In IBM Cloud Pak for Integration 2021.1.1, la licenza deve essere la licenza di prova `L-RJON-BYRMYW`
- Accettare la licenza modificando `false` in `true`.

YAML risorsa personalizzata gestore code:

```
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
```

Assicurarti di essere nello spazio dei nomi creato in precedenza, distribuisci l'YAML aggiornato, utilizzando la console web Red Hat OpenShift Container Platform , la riga di comando o utilizzando IBM Cloud Pak for Integration Platform Navigator.

Si verifica un breve ritardo mentre il sistema configura il gestore code HA nativo, dopo il quale il gestore code deve essere disponibile per l'utilizzo.

Convalida

In questa sezione viene convalidato che il gestore code funziona come previsto.

Confermare che il gestore code è in esecuzione

Il gestore code è in fase di distribuzione. Confermare che si trova nello stato `Running` prima di procedere. Ad esempio:

```
oc get qmgr nativeha-example
```

Verifica la connessione al gestore code

Per confermare che il gestore code è configurato per la comunicazione TLS unidirezionale, utilizzare le applicazioni di esempio **amqsputc** e **amqsgetc** :

Trova il nome host del gestore code

Per trovare il nome host del gestore code per l'instradamento `nativeha-example-ibm-mq-qm`, eseguire il seguente comando. Il nome host viene restituito nel campo `HOST` .

```
oc get routes nativeha-example-ibm-mq-qm
```

Specificare i dettagli del gestore code

Creare un file `CCDT.JSON` che specifichi i dettagli del gestore code. Sostituire il valore host con il nome host restituito dal passo precedente.

```
{
    "channel":
    [
        {
            "name": "HAQMCHL",
            "clientConnection":
            {
                "connection":
                [
                    {
                        "host": "<host from previous step>",
                        "port": 443
                    }
                ],
                "queueManager": "HAEXAMPLE"
            },
            "transmissionSecurity":
            {
              "cipherSpecification": "ECDHE_RSA_AES_128_CBC_SHA256"
            },
            "type": "clientConnection"
        }
   ]
}
```

Esporta variabili di ambiente

Esportare le seguenti variabili di ambiente, nel modo appropriato per il proprio sistema operativo. Queste variabili verranno lette da **amqsputc** e **amqsgetc**.

Aggiornare il percorso dei file sul sistema:

```
export MQCCDTURL='<full_path_to_file>/CCDT.JSON'
export MQSSLKEYR='<full_path_to_file>/clientkey'
```

Inserire i messaggi nella coda

Esegui il seguente comando:

```
amqsputc EXAMPLE.QUEUE HAEXAMPLE
```

Se la connessione al gestore code ha esito positivo, viene emessa la seguente risposta:

```
target queue is EXAMPLE.QUEUE
```

Inserire diversi messaggi nella coda, immettendo del testo e premendo ogni volta **Invio** .

Per terminare, premere due volte **Invio** .

Richiamare i messaggi dalla coda

Esegui il seguente comando:

```
amqsgetc EXAMPLE.QUEUE HAEXAMPLE
```

I messaggi aggiunti nel passo precedente sono stati utilizzati e vengono emessi.

Dopo pochi secondi, il comando esce.

Forza l'esito negativo del pod attivo

Per convalidare il ripristino automatico del gestore code, simula un errore pod:

Visualizza i pod attivi e in standby

Esegui il seguente comando:

```
oc get pods --selector app.kubernetes.io/instance=nativeha-example
```

Nota che, nel campo **READY** , il pod attivo restituisce il valore 1/1, mentre i pod di replica restituiscono il valore 0/1.

Elimina il pod attivo

Esegui il seguente comando, specificando il nome completo del pod attivo:

```
oc delete pod nativeha-example-ibm-mq-<value>
```

Visualizza di nuovo lo stato del pod

Esegui il seguente comando:

```
oc get pods --selector app.kubernetes.io/instance=nativeha-example
```

Visualizza lo stato del gestore code

Esegui il seguente comando, specificando il nome completo di uno degli altri pod:

```
oc exec -t Pod -- dspmq -o nativeha -x -m HAEXAMPLE
```

Dovresti vedere lo stato che mostra che l'istanza attiva è cambiata, ad esempio:

```
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)
```

Inserire e richiamare nuovamente i messaggi

Dopo che il pod standby diventa il pod attivo (ossia, dopo che il valore del campo READY diventa 1/1), utilizzare di nuovo i seguenti comandi, come descritto in precedenza, per inserire i messaggi nel gestore code, quindi richiamare i messaggi dal gestore code:

```
amqsputc EXAMPLE.QUEUE HAEXAMPLE
```

```
amqsgetc EXAMPLE.QUEUE HAEXAMPLE
```

Congratulazioni, hai correttamente distribuito un gestore code HA nativo e hai mostrato che può eseguire automaticamente il recupero da un malfunzionamento del pod.

Ulteriori informazioni

Nota 1: creazione di un instradamento

Se stai usando un IBM MQ client o un toolkit precedente a IBM MQ 9.2.1, devi creare un instradamento.

Per creare la rotta, assicurati di essere nello spazio dei nomi che hai creato precedentemente (vedi “Prima di iniziare” a pagina 94), quindi immetti il seguente YAML nella console Web di Red Hat OpenShift Container Platform o utilizzando la riga di comando:

```
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
```

Si noti che Red Hat OpenShift Container Platform Router utilizza SNI per instradare le richieste al gestore code IBM MQ . Se si modifica il nome del canale specificato nella mappa di configurazione contenente i comandi MQSC, è necessario modificare anche il campo host qui e nel file `CCDT.JSON` che specifica i dettagli del gestore code. Per ulteriori informazioni, fare riferimento a “Configurazione di un instradamento per la connessione a un gestore code dall'esterno di un cluster Red Hat OpenShift” a pagina 108.

Nota 2: utilizzo di una classe di memoria non predefinita

Questo esempio prevede che una classe di memoria predefinita sia stata configurata in Red Hat OpenShift Container Platform, pertanto non è richiesta alcuna informazione di memoria nella risorsa personalizzata del gestore code YAML. Se non si dispone di una classe di archiviazione configurata come predefinita o si desidera utilizzare una classe di archiviazione differente, aggiungere `defaultClass: <storage_class_name>` in `spec.queueManager.storage`.

Il nome della classe di archiviazione deve corrispondere esattamente al nome di una classe di archiviazione già esistente. Ciò significa che deve corrispondere al nome restituito dal comando `oc get storageclass`. Deve anche supportare `ReadWriteMany`. Per ulteriori informazioni, consultare “Considerazioni sulla memoria per IBM MQ Operator” a pagina 11.

Attività correlate

“Visualizzazione dello stato dei gestori code della HA nativa per i contenitori certificati IBM MQ” a pagina 99

Per i contenitori certificati IBM MQ , puoi visualizzare lo stato delle istanze Native HA eseguendo il comando **dspmq** all'interno di uno dei pod in esecuzione.

CP4I V9.2.2 CD *Visualizzazione dello stato dei gestori code della HA nativa per i contenitori certificati IBM MQ*

Per i contenitori certificati IBM MQ , puoi visualizzare lo stato delle istanze Native HA eseguendo il comando **dspmq** all'interno di uno dei pod in esecuzione.

Informazioni su questa attività

Importante:

È possibile utilizzare il comando **dspmq** in uno dei pod in esecuzione per visualizzare lo stato operativo di un'istanza del gestore code. Le informazioni restituite dipendono dal fatto che l'istanza sia attiva o una replica. Le informazioni fornite dall'istanza attiva sono definitive, le informazioni dai nodi di replica potrebbero non essere aggiornate.

È possibile effettuare le seguenti azioni:

- Visualizzare se l'istanza del gestore code sul nodo corrente è attiva o una replica.
- Visualizza lo stato operativo della HA nativa dell'istanza sul nodo corrente.
- Visualizzare lo stato operativo di tutte e tre le istanze in una configurazione HA nativa.

I seguenti campi di stato vengono utilizzati per riportare lo stato di configurazione della HA nativa:

Ruolo
Specifica il ruolo corrente dell'istanza ed è uno tra `Active`, `Replica`o `Unknown`.

ISTANZA
Il nome fornito per questa istanza del gestore code quando è stata creata utilizzando l'opzione **`-lr`** del comando **`crtmqm`** .

INSYNC
Indica se l'istanza è in grado di assumere il controllo come istanza attiva, se richiesto.

Quorum
Riporta lo stato del quorum nel formato *number_of_instances_in - sync/ number_of_instances_configured*.

REPLADDR
L'indirizzo di replica dell'istanza del gestore code.

COLLEGA
Indica se il nodo è connesso all'istanza attiva.

BACKLOG
Indica il numero di KB in cui si trova l'istanza.

CONNETTIN
Indica se l'istanza denominata è connessa a questa istanza.

ALTDATE
Indica la data in cui queste informazioni sono state aggiornate l'ultima volta (vuoto se non sono mai state aggiornate).

ALTTIME
Indica l'ora in cui queste informazioni sono state aggiornate l'ultima volta (vuoto se non sono mai state aggiornate).

Procedura

- Trova i pod che fanno parte del tuo gestore code.

```
oc get pod --selector app.kubernetes.io/instance=nativeha-qm
```

- Esegui il `dspmq` in uno dei pod

```
oc exec -t Pod dspmq
```

```
oc rsh Pod
```

per una shell interattiva, dove è possibile eseguire direttamente `dspmq` .

- Per determinare se un'istanza del gestore code è in esecuzione come istanza attiva o come replica:

```
oc exec -t Pod dspmq -o status -m QMgrName
```

Un'istanza attiva di un gestore code denominato BOB riporta il seguente stato:

```
QMNAME(BOB)                STATUS(Running)
```

Un'istanza di replica di un gestore code denominato BOB riporta il seguente stato:

```
QMNAME(BOB)                STATUS(Replica)
```

Un'istanza inattiva riporta il seguente stato:

```
QMNAME(BOB)                STATUS(Ended Immediately)
```

- Per determinare lo stato operativo della HA nativa dell'istanza nel pod specificato:

```
oc exec -t Pod dspmq -o nativeha -m QMgrName
```

L'istanza attiva di un gestore code denominato BOB potrebbe riportare il seguente stato:

```
QMNAME(BOB)               ROLE(Active) INSTANCE(inst1) INSYNC(Yes) QUORUM(3/3)
```

Un'istanza di replica di un gestore code denominato BOB potrebbe riportare il seguente stato:

```
QMNAME(BOB)               ROLE(Replica) INSTANCE(inst2) INSYNC(Yes) QUORUM(2/3)
```

Un'istanza inattiva di un gestore code denominato BOB potrebbe riportare il seguente stato:

```
QMNAME(BOB)               ROLE(Unknown) INSTANCE(inst3) INSYNC(no) QUORUM(0/3)
```

- Per determinare lo stato operativo della HA nativo di tutte le istanze nella configurazione della HA nativa:

```
oc exec -t Pod dspmq -o nativeha -x -m QMgrName
```

Se si immette questo comando sul nodo che esegue l'istanza attiva del BOB del gestore code, è possibile che si riceva il seguente stato:

```
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)
```

Se si immette questo comando su un nodo che esegue un'istanza di replica del BOB del gestore code, è possibile che si riceva il seguente stato, che indica che una delle repliche è in ritardo:

```
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)
```

Se si immette questo comando su un nodo che esegue un'istanza inattiva del BOB del gestore code, è possibile che si riceva il seguente stato:

```
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()
```

Se si immette il comando quando le istanze stanno ancora negoziando quali sono attive e quali sono repliche, si riceverà il seguente stato:

```
QMNAME(BOB)              STATUS(Negotiating)
```

Riferimenti correlati

comando dspmq (visualizza gestori code)

“Esempio: configurazione di un gestore code HA nativo” a pagina 94

Questo esempio mostra come distribuire un gestore code utilizzando la funzione alta disponibilità nativa in Red Hat OpenShift Container Platform (OCP) utilizzando IBM MQ Operator.

CP4I CD V 9.2.3 *Ottimizzazione avanzata per Native HA*

Impostazioni avanzate per l'ottimizzazione di intervalli e intervalli. Non dovrebbe essere necessario utilizzare queste impostazioni a meno che i valori predefiniti non corrispondano ai requisiti del sistema.

Le opzioni di base per la configurazione della HA nativa vengono gestite utilizzando l'API `QueueManager` , che IBM MQ Operator utilizza per configurare i file INI del gestore code sottostante. Ci sono alcune opzioni più avanzate che sono configurabili solo utilizzando un file INI, nella stanza NativeHALocalInstance. Consultare anche “Esempio: fornitura di file MQSC e INI” a pagina 86 per ulteriori informazioni su come configurare un file INI.

`HeartbeatInterval`
L'intervallo di heartbeat definisce la frequenza in millisecondi con cui un'istanza attiva di un gestore code HA nativo invia un heartbeat di rete. L'intervallo valido del valore dell'intervallo di heartbeat è compreso tra 500 (0.5 secondi) e 60000 (1 minuto), un valore esterno a questo intervallo causa l'errore di avvio del gestore code. Se questo attributo viene omesso, viene utilizzato un valore predefinito di 5000 (5 secondi). Ogni istanza deve utilizzare lo stesso intervallo di heartbeat.

`HeartbeatTimeout`
Il timeout heartbeat definisce il tempo di attesa di un'istanza di replica di un gestore code HA nativo prima di decidere che l'istanza attiva non risponde. L'intervallo valido del valore di timeout dell'intervallo di heartbeat è compreso tra 500 (0.5 secondi) e 120000 (2 minuti). Il valore del timeout di heartbeat deve essere maggiore o uguale all'intervallo di heartbeat.

Un valore non valido causa l'errore di avvio del gestore code. Se questo attributo viene omesso, una replica attende 2 x `HeartbeatInterval` prima di avviare il processo per selezionare una nuova istanza attiva. Ogni istanza deve utilizzare lo stesso timeout heartbeat.

`RetryInterval`
L'intervallo di nuovi tentativi definisce la frequenza in millisecondi con cui un gestore code HA nativo deve ritentare un link di replica non riuscito. L'intervallo valido per i tentativi è compreso tra 500 (0.5 secondi) e 120000 (2 minuti). Se questo attributo viene omesso, una replica attende 2 x `HeartbeatInterval` prima di ritentare un link di replica non riuscito.

CP4I *Chiusura gestori code della HA nativa*
È possibile utilizzare il comando **endmqm** per terminare un gestore code attivo o di replica che fa parte di un gruppo HA nativo.

Procedura

- Per terminare l''istanza attiva di un gestore code, fare riferimento a Fine dei gestori code della HA nativa nella sezione Configurazione di questa documentazione.

CP4I V 9.2.2 CD *Valutazione della funzione HA nativa in IBM Cloud Pak for Integration 2021.1.1*
Il periodo di valutazione IBM Cloud Pak for Integration 2021.1.1 Native HA è terminato. Utilizzare la funzione Native HA aggiornata disponibile all'indirizzo IBM Cloud Pak for Integration 2021.2.1, utilizzando IBM MQ Operator 1.6 o superiore con IBM MQ 9.2.3 o superiore.

Attività correlate
“Visualizzazione dello stato dei gestori code della HA nativa per i contenitori certificati IBM MQ” a pagina 99
Per i contenitori certificati IBM MQ , puoi visualizzare lo stato delle istanze Native HA eseguendo il comando **dspmq** all'interno di uno dei pod in esecuzione.

Riferimenti correlati
“Esempio: configurazione di un gestore code HA nativo” a pagina 94
Questo esempio mostra come distribuire un gestore code utilizzando la funzione alta disponibilità nativa in Red Hat OpenShift Container Platform (OCP) utilizzando IBM MQ Operator.

OpenShift CP4I ***Esempio: configurazione di un gestore code a più istanze***
Questo esempio mostra come distribuire un gestore code a più istanze in Red Hat OpenShift Container Platform (OCP) utilizzando IBM MQ Operator. In questo esempio, si configura anche la comunicazione TLS unidirezionale tra un client di esempio e il gestore code. L'esempio dimostra la corretta configurazione inserendo e ottenendo i messaggi prima e dopo un errore del pod simulato.

Prima di iniziare

Per completare questo esempio, è necessario prima aver completato i prerequisiti riportati di seguito:

- Installare IBM MQ cliente aggiungere le directory `samp/bin` e `bin` installate al *PATH*. Il client fornisce le applicazioni **runmqakm**, **amqsputc** e **amqsgetc** richieste da questo esempio. Installare IBM MQ client nel modo seguente:
  - Linux Windows Per Windows e Linux: installare il client ridistribuibile IBM MQ per il proprio sistema operativo da https://ibm.biz/mq92redistclients
  - macOS Per Mac: scaricare e configurare IBM MQ MacOS Toolkit. Vedere https://developer.ibm.com/tutorials/mq-macos-dev/.
- Installare lo strumento OpenSSL per il proprio sistema operativo. È necessario per generare un certificato autofirmato per il gestore code, se non si dispone già di una chiave privata e di un certificato.
- Creare un progetto / spazio dei nomi Red Hat OpenShift Container Platform (OCP) per questo esempio.
- Sulla riga comandi, accedere al cluster OCP e passare allo spazio dei nomi precedente.
- Assicurarsi che il file IBM MQ Operator sia installato e disponibile nello spazio dei nomi precedente.
- Configurare una classe di memoria predefinita in OCP, che deve essere utilizzata dal proprio gestore code. Se vuoi completare questa esercitazione senza impostare una classe di archiviazione predefinita, vedi Nota 2: Utilizzo di una classe di archiviazione non predefinita.

Informazioni su questa attività

I gestori code a più istanze implicano un pod Kubernetes attivo e uno standby. Questi vengono eseguiti come parte di una serie con stato Kubernetes con esattamente due repliche e una serie di volumi persistenti Kubernetes . Per ulteriori informazioni sui gestori code a più istanze, consultare “Alta disponibilità per IBM MQ nei contenitori” a pagina 16.

L'esempio fornisce una risorsa personalizzata YAML che definisce un gestore code a più istanze con archiviazione persistente e configurato con TLS. Dopo aver distribuito il gestore code in OCP, si simula l'errore del pod del gestore code attivo. Si vede che si verifica un ripristino automatico e si prova che ha avuto esito positivo inserendo e ricevendo messaggi dopo l'errore.

Esempio

Creare una chiave privata TLS e i certificati per il server MQ

In questa sezione viene descritto come creare un certificato autofirmato per il gestore code e come aggiungere il certificato ad un database di chiavi per agire come truststore per il client. Se si dispone già di una chiave privata e di un certificato, è possibile utilizzarli. Notare che è necessario utilizzare i certificati autofirmati solo per scopi di sviluppo.

Per creare una chiave privata autofirmata e un certificato pubblico nella directory corrente, esegui questo comando:

```
openssl req -newkey rsa:2048 -nodes -keyout tls.key -subj "/CN=localhost" -x509 -days 3650
-out tls.crt
```

Aggiungi la chiave pubblica del gestore code a un database di chiavi del client

Un database di chiavi client viene utilizzato come truststore per l'applicazione client.

Creare il database delle chiavi client:

```
runmqakm -keydb -create -db clientkey.kdb -pw password -type cms -stash
```

Aggiungere la chiave pubblica generata in precedenza al database delle chiavi client:

```
runmqakm -cert -add -db clientkey.kdb -label mqservercert -file tls.crt -format ascii
-stashed
```

Crea un segreto contenente i certificati TLS per la distribuzione del gestore code

In modo che il tuo gestore code possa fare riferimento e applicare la chiave e il certificato, creare un segreto TLS Kubernetes , facendo riferimento ai file creati in precedenza. Quando si esegue questa operazione, assicurarsi di essere nello spazio dei nomi creato prima di iniziare questa attività.

```
oc create secret tls example-mi-secret --key="tls.key" --cert="tls.crt"
```

Crea una mappa di configurazione contenente i comandi MQSC

Crea una mappa di configurazione Kubernetes contenente i comandi MQSC per creare una nuova coda e un canale SVRCONN e per aggiungere un record di autenticazione di canale che consenta l'accesso al canale bloccando solo gli utenti denominati *nobody*.

Notare che questo approccio deve essere utilizzato solo per scopi di sviluppo.

Assicurarsi di essere nello spazio dei nomi creato in precedenza (consultare “Prima di iniziare” a pagina 103), quindi immettere il seguente YAML nell'interfaccia utente OCP o utilizzando la riga comandi:

```
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)
```

Configura instradamento

Se si sta utilizzando un IBM MQ client o un toolkit all'indirizzo IBM MQ 9.2.1 o successivo, è possibile configurare l'instradamento al gestore code utilizzando un file di configurazione del gestore code (un file INI). All'interno del file, impostare la variabile *OutboundSNI* per l'instradamento in base al nome host anziché al nome del canale.

Creare un file nella directory in cui si stanno eseguendo i comandi, denominata `mqclient.ini`, contenente il testo seguente:

```
#* 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
```

Nota: non modificare alcun valore in questa pagina. Ad esempio, la stringa HOSTNAME deve essere lasciata così com' è.

Per ulteriori dettagli, consultare Stanza SSL del file di configurazione client.

Se si utilizza un IBM MQ client o un toolkit precedente a IBM MQ 9.2.1, è necessario creare un instradamento OCP invece del file di configurazione precedente. Segui la procedura in Nota 1: creazione di un instradamento.

Distribuisci il gestore code

Importante: in questo esempio utilizziamo la variabile *MQSNOAUT* per disabilitare l'autorizzazione sul gestore code, che ci consente di concentrarci sui passi richiesti per connettere un client utilizzando TLS. Ciò non è consigliato in una distribuzione di produzione di IBM MQ, perché fa sì che tutte le applicazioni che si collegano abbiano pieni poteri di gestione, senza alcun meccanismo per ridurre le autorizzazioni per le singole applicazioni.

Copiare e aggiornare il seguente YAML.

- Assicurarsi che sia stata specificata la licenza corretta. Consultare Riferimento alle licenze per mq.ibm.com/v1beta1.
- Accettare la licenza modificando `false` in `true`.
- Se si utilizza IBM Cloud File Storage, consultare Nota 3: Utilizzo di IBM Cloud File Storage

YAML risorsa personalizzata gestore code:

```
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
```

Assicurarti di essere nello spazio dei nomi che hai creato in precedenza, distribuire lo YAML aggiornato nell'IU OCP, utilizzando la riga di comando o utilizzando il IBM Cloud Pak for Integration Platform Navigator.

Convalida

Dopo un breve ritardo, il gestore code a più istanze deve essere configurato e disponibile per l'utilizzo. In questa sezione viene convalidato che il gestore code funziona come previsto.

Confermare che il gestore code è in esecuzione

Il gestore code è in fase di distribuzione. Confermare che si trova nello stato `Running` prima di procedere. Ad esempio:

```
oc get qmgr miexample
```

Verifica la connessione al gestore code

Per confermare che il gestore code è configurato per la comunicazione TLS unidirezionale, utilizzare le applicazioni di esempio **amqsputc** e **amqsgetc** :

Trova il nome host del gestore code

Per trovare il nome host del gestore code per l'instradamento `miexample-ibm-mq-qm`, eseguire il seguente comando. Il nome host viene restituito nel campo `HOST` .

```
oc get routes miexample-ibm-mq-qm
```

Specificare i dettagli del gestore code

Creare un file `CCDT.JSON` che specifichi i dettagli del gestore code. Sostituire il valore host con il nome host restituito dal passo precedente.

```
{
    "channel":
    [
        {
            "name": "MIQMCHL",
            "clientConnection":
            {
                "connection":
                [
                    {
                        "host": "<host from previous step>",
                        "port": 443
                    }
                ],
                "queueManager": "MIEXAMPLE"
            },
            "transmissionSecurity":
            {
              "cipherSpecification": "ECDHE_RSA_AES_128_CBC_SHA256"
            },
            "type": "clientConnection"
        }
   ]
}
```

Esporta variabili di ambiente

Esportare le seguenti variabili di ambiente, nel modo appropriato per il proprio sistema operativo. Queste variabili verranno lette da **amqsputc** e **amqsgetc**.

Aggiornare il percorso dei file sul sistema:

```
export MQCCDTURL='<full_path_to_file>/CCDT.JSON'
export MQSSLKEYR='<full_path_to_file>/clientkey'
```

Inserire i messaggi nella coda

Esegui il seguente comando:

```
amqsputc EXAMPLE.QUEUE MIEXAMPLE
```

Se la connessione al gestore code ha esito positivo, viene emessa la seguente risposta:

```
target queue is EXAMPLE.QUEUE
```

Inserire diversi messaggi nella coda, immettendo del testo e premendo ogni volta **Invio** .

Per terminare, premere due volte **Invio** .

Richiamare i messaggi dalla coda

Esegui il seguente comando:

```
amqsgetc EXAMPLE.QUEUE MIEXAMPLE
```

I messaggi aggiunti nel passo precedente sono stati utilizzati e vengono emessi.

Dopo pochi secondi, il comando esce.

Forza l'esito negativo del pod attivo

Per convalidare il ripristino automatico del gestore code a più istanze, simula un errore pod:

Visualizza i pod attivi e in standby

Esegui il seguente comando:

```
oc get pods
```

Tieni presente che, nel campo **READY** , il pod attivo restituisce il valore 1/1, mentre il pod standby restituisce il valore 0/1.

Elimina il pod attivo

Esegui il seguente comando, specificando il nome completo del pod attivo:

```
oc delete pod miexample-ibm-mq-<value>
```

Visualizza di nuovo lo stato del pod

Esegui il seguente comando:

```
oc get pods
```

Visualizza il log del pod standby

Eseguire il seguente comando, specificando il nome completo del pod standby:

```
oc logs miexample-ibm-mq-<value>
```

Dovrebbe essere visualizzato il seguente messaggio:

```
IBM MQ queue manager 'MIEXAMPLE' becoming the active instance.
```

Inserire e richiamare nuovamente i messaggi

Dopo che il pod standby diventa il pod attivo (ossia, dopo che il valore del campo READY diventa 1/1), utilizzare di nuovo i seguenti comandi, come descritto in precedenza, per inserire i messaggi nel gestore code, quindi richiamare i messaggi dal gestore code:

```
amqsputc EXAMPLE.QUEUE MIEXAMPLE
```

```
amqsgetc EXAMPLE.QUEUE MIEXAMPLE
```

Congratulazioni, hai correttamente distribuito un gestore code a più istanze e hai mostrato che può eseguire automaticamente il ripristino da un errore del pod.

Ulteriori informazioni

Nota 1: creazione di un instradamento

Se si utilizza un IBM MQ client o un toolkit precedente a IBM MQ 9.2.1, è necessario creare un instradamento OCP.

Per creare la rotta, assicurati di essere nello spazio dei nomi che hai creato in precedenza (vedi “Prima di iniziare” a pagina 103), quindi immetti il seguente YAML nell'IU OCP o utilizzando la riga di comando:

```
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
```

Si noti che Red Hat OpenShift Container Platform Router utilizza SNI per instradare le richieste al gestore code IBM MQ . Se si modifica il nome del canale specificato nella mappa di configurazione contenente i comandi MQSC, è necessario modificare anche il campo host qui e nel file CCDT.JSON che specifica i dettagli del gestore code. Per ulteriori informazioni, fare riferimento a “Configurazione

di un instradamento per la connessione a un gestore code dall'esterno di un cluster Red Hat OpenShift” a pagina 108.

Nota 2: utilizzo di una classe di memoria non predefinita

Questo esempio prevede che una classe di archiviazione predefinita sia stata configurata in OCP, pertanto non sono richieste informazioni di archiviazione nell' YAML della risorsa personalizzata del gestore code. Se non si dispone di una classe di archiviazione configurata come predefinita o si desidera utilizzare una classe di archiviazione differente, aggiungere `defaultClass: <storage_class_name>` in `spec.queueManager.storage`.

Il nome della classe di memoria deve corrispondere esattamente al nome di una classe di memoria esistente sul sistema OCP. Ciò significa che deve corrispondere al nome restituito dal comando `oc get storageclass`. Deve anche supportare `ReadWriteMany`. Per ulteriori informazioni, consultare “Considerazioni sulla memoria per IBM MQ Operator” a pagina 11.

Nota 3: utilizzo IBM Cloud File Storage

In alcune situazioni, ad esempio quando si utilizza IBM Cloud File Storage, è necessario anche specificare il campo **`securityGroups`** nell' YAML della risorsa personalizzata del gestore code. Ad esempio, aggiungendo il seguente campo secondario direttamente sotto `spec:`

```
securityContext:
  supplementalGroups: [99]
```

Per ulteriori informazioni, consultare “Considerazioni sulla memoria per IBM MQ Operator” a pagina 11.

OpenShift CP4I CD Configurazione di un instradamento per la connessione a un gestore code dall'esterno di un cluster Red Hat OpenShift

Hai bisogno di un instradamento Red Hat OpenShift per connettere un'applicazione a un gestore code IBM MQ dall'esterno di un cluster Red Hat OpenShift . È necessario abilitare TLS sul gestore code e sull'applicazione client IBM MQ , perché SNI è disponibile solo nel protocollo TLS quando viene utilizzato un protocollo TLS 1.2 o superiore. Red Hat OpenShift Container Platform Router utilizza SNI per instradare le richieste al gestore code IBM MQ .

Informazioni su questa attività

Attenzione: Questo documento si applica alle versioni 9.2.1 Continuous Delivery e successive dei client IBM MQ . Se il client utilizza la versione 9.2.0 Long Term Support o precedente, fare riferimento alla IBM MQ 9.1 pagina della documentazione Connessione a un gestore code distribuito in un Red Hat OpenShift cluster.

V 9.2.1 La configurazione richiesta dell' Red Hat OpenShift instradamento dipende dal comportamento SNI (Server Name Indication) della propria applicazione client. IBM MQ supporta due differenti impostazioni di intestazione SNI a seconda del tipo di configurazione e client. Un'intestazione SNI è impostata sul nome host della destinazione del client o, in alternativa, sul nome del canale IBM MQ . Per informazioni sul modo in cui IBM MQ associa un nome canale a un nome host, vedi How IBM MQ fornisce più capacità di certificati.

V 9.2.1 Se un'intestazione SNI è impostata su un nome di canale IBM MQ o se un nome host è controllato utilizzando l'attributo **`OutboundSNI`** . I valori possibili sono `OutboundSNI=CHANNEL` (il valore predefinito) o `OutboundSNI=HOSTNAME`. Per ulteriori informazioni, consultare Stanza SSL del file di configurazione client. Si noti che `CHANNEL` e `HOSTNAME` sono i valori esatti che si utilizzano; non sono nomi di variabili che si sostituiscono con un nome canale o un nome host effettivo.

V 9.2.1

Comportamenti client con impostazioni `OutboundSNI` differenti

Se **`OutboundSNI`** è impostato su `HOSTNAME`, i seguenti client impostano un nome host SNI purché venga fornito un nome host nel nome connessione:

- Client C
- Client .NET in modalità non gestita
- Java/JMS Client

Se **OutboundSNI** è impostato su HOSTNAME e viene utilizzato un indirizzo IP nel nome della connessione, i seguenti client inviano un'intestazione SNI vuota:

- Client C
- Client .NET in modalità non gestita
- Java/JMS Client (che non possono eseguire una ricerca DNS inversa del nome host)

Se **OutboundSNI** è impostato su CHANNELo non è impostato, viene utilizzato un nome canale IBM MQ e viene sempre inviato, se viene utilizzato un nome host o un nome connessione indirizzo IP.

I tipi di client seguenti non supportano l'impostazione di un'intestazione SNI su un nome canale IBM MQ e quindi tentano sempre di impostare l'intestazione SNI su un nome host indipendentemente dall'impostazione **OutboundSNI** :

- Client AMQP
- Client XR
- Client .NET in modalità gestita (Prima IBM MQ 9.2.0 Fix Pack 4 per Long Term Support e prima IBM MQ 9.2.3 per Continuous Delivery.)

V 9.2.0.4 V 9.2.3

Da IBM MQ 9.2.0 Fix Pack 4 per Long Term Support e IBM MQ 9.2.3 per Continuous Delivery, il client IBM MQ gestito .NET è stato aggiornato per impostare SERVERNAME sul rispettivo nome host se la proprietà **OutboundSNI** è impostata su HOSTNAME, il che consente a un client IBM MQ gestito .NET di connettersi a un gestore code utilizzando gli instradamenti Red Hat OpenShift . Tenere presente che, in IBM MQ 9.2.0 Fix Pack 4, la proprietà **OutboundSNI** viene aggiunta e supportata solo dal file `mqclient.ini` ; non è possibile impostare la proprietà dall'applicazione .NET.

V 9.2.5

Se un'applicazione client si connette a un gestore code distribuito in un cluster Red Hat OpenShift tramite IBM MQ Internet Pass-Thru (MQIPT), MQIPT può essere configurato per impostare SNI sul nome host utilizzando la proprietà SSLClientOutboundSNI nella definizione di instradamento.

OutboundSNI, più certificati e instradamenti Red Hat OpenShift

IBM MQ utilizza l'intestazione SNI per fornire più funzionalità di certificati. Se un'applicazione si connette a un canale IBM MQ configurato per utilizzare un certificato differente tramite il campo CERTLABL, l'applicazione deve connettersi con un'impostazione **OutboundSNI** di CHANNEL.

Se la configurazione dell'instradamento Red Hat OpenShift richiede un HOSTNAME SNI, non è possibile utilizzare la funzionalità di più certificati di IBM MQ e non è possibile impostare un'impostazione CERTLABL su qualsiasi oggetto del canale IBM MQ .

Se un'applicazione con un'impostazione **OutboundSNI** diversa da CHANNEL si connette ad un canale con un'etichetta di certificato configurata, l'applicazione viene rifiutata con un MQRC_SSL_INITIALIZATION_ERROR e un messaggio AMQ9673 viene stampato nei log degli errori del gestore code.

Per ulteriori informazioni su come IBM MQ fornisce la funzionalità di più certificati, consultare Come IBM MQ fornisce la funzionalità di più certificati .

Esempio

Le applicazioni client che impostano SNI sul canale MQ richiedono la creazione di un nuovo instradamento Red Hat OpenShift per ogni canale a cui si desidera connettersi. È inoltre necessario utilizzare nomi canale univoci nel cluster Red Hat OpenShift Container Platform , per consentire l'instradamento al gestore code corretto.

È importante che i nomi dei canali MQ non terminino con una lettera minuscola a causa del modo in cui IBM MQ associa i nomi dei canali alle intestazioni SNI.

Per stabilire il nome host richiesto per ciascuno dei tuoi nuovi instradamenti Red Hat OpenShift , devi associare ciascun nome di canale a un indirizzo SNI. Per ulteriori informazioni, vedi How IBM MQ fornisce più capacità di certificati .

Devi quindi creare un nuovo instradamento Red Hat OpenShift per ciascun canale, applicando il seguente `yaml` nel tuo 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
```

Configurazione dei dettagli di connessione dell'applicazione client

È possibile stabilire il nome host da utilizzare per la connessione client immettendo il seguente comando:

```
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}"
```

La porta per la connessione client deve essere impostata sulla porta utilizzata dal router Red Hat OpenShift Container Platform - normalmente 443.

Attività correlate

“Connessione al IBM MQ Console distribuito in un cluster Red Hat OpenShift” a pagina 115
Modalità di connessione a IBM MQ Console di un gestore code distribuito su un cluster Red Hat OpenShift Container Platform .

CP4I Integrazione con IBM Cloud Pak for Integration Operations Dashboard

La capacità di tracciare le transazioni tramite IBM Cloud Pak for Integration è fornita dal dashboard Operazioni.

Informazioni su questa attività

L'abilitazione dell'integrazione con Operations Dashboard installa un'uscita dell'API MQ nel gestore code. L'uscita API invierà i dati di traccia all'archivio dati di Operations Dashboard, relativi ai messaggi che passano attraverso il gestore code.

Tenere presente che viene eseguita la traccia solo dei messaggi inviati utilizzando i collegamenti client MQ .

Notare inoltre che per le versioni di IBM MQ Operator precedenti a 1.5, quando la traccia è abilitata, le immagini dell'agent di traccia e del raccoglitore distribuite insieme al gestore code erano sempre le ultime versioni disponibili, il che potrebbe introdurre un'incompatibilità se non si utilizza la versione più recente di IBM Cloud Pak for Integration.

Procedura

1. Distribuisci un gestore code con la traccia abilitata

   Per impostazione predefinita, la funzione di traccia è disabilitata.

Se si sta eseguendo la distribuzione utilizzando IBM Cloud Pak for Integration Platform Navigator, è possibile abilitare la traccia durante la distribuzione, impostando **Abilita traccia** su **On**e impostando **Spazio dei nomi di traccia** sullo spazio dei nomi in cui è installato Operations Dashboard. Per ulteriori informazioni sulla distribuzione di un gestore code, consultare “Distribuzione di un gestore code utilizzando IBM Cloud Pak for Integration Platform Navigator” a pagina 81

Se stai eseguendo la distribuzione utilizzando la CLIRed Hat OpenShift o la console webRed Hat OpenShift, puoi abilitare la traccia con il seguente frammento YAML:

```
spec:
  tracing:
    enabled: true
    namespace: <Operations_Dashboard_Namespace
```

Importante: il gestore code non verrà avviato fino a quando MQ non sarà stato registrato con il dashboard Operazioni (consultare il passo successivo).

Tenere presente che quando questa funzione è abilitata, eseguirà due contenitori sidecar ("Agent" e "Collector") in aggiunta al contenitore del gestore code. Le immagini per questi contenitori laterali saranno disponibili nello stesso registro dell'immagine MQ principale e utilizzeranno la stessa politica di pull e il segreto di pull. Sono disponibili ulteriori impostazioni per configurare i limiti di CPU e memoria.

2. Se questa è la prima volta che un gestore code con l'integrazione di Operations Dashboard viene distribuito in questo spazio dei nomi, è necessario Registrare con Operations Dashboard.

   La registrazione crea un oggetto segreto che il pod del gestore code deve avviare correttamente.

CP4I CD *Distribuzione o aggiornamento di IBM MQ 9.2.2 o 9.2.3 con l'integrazione di Operations Dashboard in IBM Cloud Pak for Integration 2021.4*

Ogni versione di IBM MQ è associata a una versione specifica dei componenti del raccoglitore e dell'agent di Operations Dashboard, che sono distribuiti insieme a un gestore code. IBM Cloud Pak for Integration 2021.4.1 introduce una modifica che fa sì che i vecchi componenti dell'agent e del raccoglitore non funzionino con il dashboard Operazioni. Per risolvere questo problema, è necessario sovrascrivere la versione delle immagini del raccoglitore e dell'agent del dashboard Operazioni che si utilizzano, quando si utilizza IBM MQ 9.2.2 o 9.2.3.

Distribuzione di un nuovo gestore code IBM MQ 9.2.2 o 9.2.3

Quando si utilizza IBM Cloud Pak for Integration 2021.4.1 con IBM MQ 9.2.2 o 9.2.3, è necessario sovrascrivere le immagini del raccoglitore e dell'agent Operations Dashboard alle versioni 2.4 nel `QueueManager` YAML. Ad esempio:

```
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
```

Se non si esegue questa operazione, il pod `QueueManager` verrà bloccato nello stato `Pending` . Quando si passa a IBM MQ 9.2.4, è possibile rimuovere queste sovrascritture.

Aggiornamento a IBM Cloud Pak for Integration 2021.4.1

Nota: Se si sta conservando il gestore code IBM MQ 9.2.2 o 9.2.3 , non completare il passo 3.

1. Aggiornare `QueueManager` per sovrascrivere le immagini dell'agente e del programma di raccolta, come precedentemente descritto.
2. Aggiornare gli operatori IBM Cloud Pak for Integration , inclusi il dashboard Operazioni e l'operatore IBM MQ , come descritto in “Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 71.

3. (Facoltativo) Per eseguire l'aggiornamento a IBM MQ 9.2.4 o versioni successive, aggiornare `QueueManager` per utilizzare `.spec.version` per la propria versione di IBM MQ, quindi rimuovere la sovrascrittura delle immagini dell'agent e del raccoglitore.

OpenShift CP4I Creazione di un'immagine con file MQSC e INI personalizzati, utilizzando la CLI Red Hat OpenShift

Utilizzare una pipeline Red Hat OpenShift Container Platform per creare una nuova immagine contenitore IBM MQ , con i file MQSC e INI che si desidera applicare ai gestori code che utilizzano questa immagine. Questa attività deve essere completata da un amministratore del progetto

Prima di iniziare

È necessario installare la CLI (command - line interface) Red Hat OpenShift Container Platform.

Accedi al tuo cluster utilizzando **`cloudctl login`** (per IBM Cloud Pak for Integration) o **`oc login`**.

Se non hai un segreto Red Hat OpenShift per IBM Entitled Registry nel tuo progetto Red Hat OpenShift , attieniti alla procedura per “Preparazione del tuo progetto Red Hat OpenShift per IBM MQ” a pagina 79.

Procedura

1. Crea un `ImageStream`

   Un flusso di immagini e i tag associati forniscono un'astrazione per fare riferimento alle immagini del contenitore da Red Hat OpenShift Container Platform. Il flusso di immagini e le relative tag consentono di vedere quali immagini sono disponibili e di verificare che si stia utilizzando l'immagine specifica necessaria anche se l'immagine nel repository cambia.

   ```
   oc create imagestream mymq
   ```

2. Crea un `BuildConfig` per la nuova immagine

   Un `BuildConfig` consentirà le build per la tua nuova immagine, che si baserà sulle immagini ufficiali di IBM , ma aggiungerà tutti i file MQSC o INI che vuoi eseguire all'avvio del contenitore.

   a) Creare un file YAML che definisca la risorsa `BuildConfig`

   Ad esempio, creare un file denominato "mq-build-config.yaml" con il seguente contenuto:

   ```
   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'
   ```

   Sarà necessario sostituire le due posizioni in cui viene menzionato il IBM MQ di base, in modo da puntare all'immagine di base corretta per la versione e la fix che si desidera utilizzare (consultare “Cronologia delle release per IBM MQ Operator” a pagina 20 per i dettagli). Man mano che le correzioni vengono applicate, sarà necessario ripetere questi passi per ricreare l'immagine.

Questo esempio crea una nuova immagine basata sull'immagine ufficiale IBM e aggiunge i file denominati "my.mqsc" e "my.ini" nella directory `/etc/mqm` . Tutti i file MQSC o INI trovati in questa directory verranno applicati dal container all'avvio. I file INI vengono applicati utilizzando l'opzione **`crtmqm -ii`** e uniti ai file INI esistenti. I file MQSC vengono applicati in ordine alfabetico.

È importante che i comandi MQSC siano ripetibili, poiché verranno eseguiti *ad ogni* avvio del gestore code. Ciò in genere significa aggiungere il parametro `REPLACE` su qualsiasi comando `DEFINE` e aggiungere il parametro `IGNSTATE(YES)` a qualsiasi comando `START` o `STOP` .

b) Applicare il `BuildConfig` al server.

```
oc apply -f mq-build-config.yaml
```

3. Esegui una build per creare la tua immagine

a) Avvia la build

```
oc start-build mymq
```

L'output dovrebbe essere simile al seguente:

```
build.build.openshift.io/mymq-1 started
```

b) Verificare lo stato della build
Ad esempio, è possibile eseguire il seguente comando, utilizzando l'identificativo di build restituito nel passaggio precedente:

```
oc describe build mymq-1
```

4. Distribuisci un gestore code, utilizzando la nuova immagine

Segui la procedura descritta in “Distribuzione di un gestore code su un cluster Red Hat OpenShift Container Platform” a pagina 81, aggiungendo la nuova immagine personalizzata in YAML.

Puoi aggiungere il seguente frammento di YAML nel tuo normale `QueueManager` YAML, dove *my - namespace* è il Red Hat OpenShift progetto/namespace che stai utilizzando e *image* è il nome dell'immagine che hai creato in precedenza (ad esempio, "mymq:latest-amd64"):

```
spec:
  queueManager:
    image: image-registry.openshift-image-registry.svc:5000/my-namespace/my-image
```

Attività correlate

“Distribuzione di un gestore code su un cluster Red Hat OpenShift Container Platform” a pagina 81
Utilizzare la risorsa personalizzata QueueManager per distribuire un gestore code su un cluster Red Hat OpenShift Container Platform.

OpenShift CP4I Aggiunta di annotazioni ed etichette personalizzate alle risorse del gestore code

Aggiungere annotazioni ed etichette personalizzate ai metadati di QueueManager .

Informazioni su questa attività

Le annotazioni e le etichette personalizzate vengono aggiunte a tutte le risorse tranne le PVC. Se un'annotazione o un'etichetta personalizzata corrisponde a una chiave esistente, viene utilizzato il valore impostato da IBM MQ Operator .

Procedura

- Aggiungere annotazioni personalizzate.

Per aggiungere annotazioni personalizzate alle risorse del gestore code, incluso il pod, aggiungi le annotazioni in `metadata`. Ad esempio:

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: quickstart-cp4i
  annotations:
    annotationKey: "value"
```

- Aggiungere etichette personalizzate.

  Per aggiungere etichette personalizzate alle risorse del gestore code, incluso il pod, aggiungere le etichette in `metadata`. Ad esempio:

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: quickstart-cp4i
  labels:
    labelKey: "value"
```

OpenShift CP4I Disabilitazione dei controlli webhook di runtime

I controlli del webhook di runtime garantiscono che le classi di memoria siano valide per il tuo gestore code. È possibile disabilitarli per migliorare le prestazioni o perché non sono validi per il proprio ambiente.

Informazioni su questa attività

I controlli webhook di runtime vengono eseguiti sulla configurazione del gestore code. Essi verificano che le classi di memoria siano adatte per il tipo di gestore code selezionato.

È possibile scegliere di disabilitare questi controlli per diminuire il tempo impiegato per la creazione del gestore code o perché i controlli non sono validi per il proprio ambiente specifico.

Nota: Dopo aver disabilitato i controlli webhook di runtime, sono consentiti tutti i valori della classe di archiviazione. Ciò potrebbe causare un gestore code interrotto.

Il supporto per i controlli di runtime è stato introdotto in IBM MQ Operator 1.2.

Procedura

- Disabilita i controlli webhook di runtime.

  Aggiungere la seguente annotazione in `metadata`. Ad esempio:

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: quickstart-cp4i
  annotations:
    "com.ibm.cp4i/disable-webhook-runtime-checks" : "true"
```

OpenShift CP4I Utilizzo di IBM MQ mediante IBM MQ Operator

Informazioni su questa attività

Procedura

- “Preparazione del tuo progetto Red Hat OpenShift per IBM MQ” a pagina 79.
- “Distribuzione di un gestore code su un cluster Red Hat OpenShift Container Platform” a pagina 81.

OpenShift CP4I Connessione al IBM MQ Console distribuito in un cluster Red Hat OpenShift

Modalità di connessione a IBM MQ Console di un gestore code distribuito su un cluster Red Hat OpenShift Container Platform .

Informazioni su questa attività

L'URL IBM MQ Console è disponibile nella pagina dei dettagli `QueueManager` nella console Web Red Hat OpenShift o in IBM Cloud Pak for Integration Platform Navigator. In alternativa, è possibile trovarlo dalla CLI Red Hat OpenShift immettendo il seguente comando:

```
oc get queuemanager <QueueManager Name> -n <namespace of your MQ deployment> --output
jsonpath='{.status.adminUiUrl}'
```

Se stai usando una licenza IBM Cloud Pak for Integration , la console web IBM MQ è configurata per utilizzare IBM Cloud Pak Identity and Access Manager (IAM). Il componente IAM potrebbe essere già stato configurato dall'amministratore del cluster. Tuttavia, se questa è la prima volta che IAM viene utilizzato sul tuo cluster Red Hat OpenShift , dovrai richiamare la password amministratore iniziale. Per ulteriori informazioni, vedi Acquisizione della password amministratore iniziale.

Se si utilizza una licenza IBM MQ , la console web di MQ non è preconfigurata ed è necessario configurarla da soli. Per ulteriori informazioni, consultare Configurazione di utenti e ruoli.

Attività correlate

“Configurazione di un instradamento per la connessione a un gestore code dall'esterno di un cluster Red Hat OpenShift” a pagina 108
Hai bisogno di un instradamento Red Hat OpenShift per connettere un'applicazione a un gestore code IBM MQ dall'esterno di un cluster Red Hat OpenShift . È necessario abilitare TLS sul gestore code e sull'applicazione client IBM MQ , perché SNI è disponibile solo nel protocollo TLS quando viene utilizzato un protocollo TLS 1.2 o superiore. Red Hat OpenShift Container Platform Router utilizza SNI per instradare le richieste al gestore code IBM MQ .

OpenShift CP4I *Concessione delle autorizzazioni per IBM MQ Console utilizzando IBM Cloud Pak IAM*

Le autorizzazioni per IBM MQ Console sono gestite tramite IBM Cloud Pak Administration Hub e non tramite IBM Cloud Pak for Integration Platform Navigator. IBM MQ non utilizza le autorizzazioni "Automation" fornite da IBM Cloud Pak for Integration, ma utilizza invece le autorizzazioni di base abilitate da IBM Cloud Pak Identity and Access Manager (IAM).

Procedura

1. Aprire la console di gestione IBM Cloud Pak .

   Dal IBM Cloud Pak for Integration Platform UI, fai clic sullo switch Cloud Pak (icona a 9 punti) nell'angolo in alto a destra della barra degli strumenti, quindi fai clic sul pannello **IBM Cloud Pak Administration** .
2. Nel menu di navigazione nell'angolo in alto a sinistra, seleziona **Identity and access**e seleziona **Teams and services IDs**.
3. Creare un team, quindi aggiungervi utenti.
   a) Selezionare **Crea team**.
   b) Immettere un nome team, quindi selezionare il dominio di sicurezza per gli utenti che si desidera gestire.
   c) Ricerca utenti.

      Questi utenti devono già esistere nel tuo provider di identità.
   d) Quando si trova ogni utente, assegnare loro un ruolo. Deve essere "Amministratore" o "Amministratore cluster", per amministrare IBM MQ utilizzando IBM MQ Console.

4. Aggiungere ogni utente a un namespace.
   a) Selezionare il team da modificare.
   b) Selezionare **Risorse** > **Gestisci risorse**.
   c) Selezionare gli spazi dei nomi che si desidera siano gestiti da questo team. Questi possono essere qualsiasi spazio dei nomi con un gestore code.

OpenShift CP4I Monitoraggio quando si utilizza IBM MQ Operator

I gestori code gestiti da IBM MQ Operator possono produrre metriche compatibili con Prometheus.

Puoi visualizzare queste metriche utilizzando lo stack di monitoraggioRed Hat OpenShift Container Platform (OCP). Aprire la scheda **Metriche** in OCP, quindi fare clic su **Observe** > **Metriche**. Le metriche del gestore code sono abilitate per impostazione predefinita, ma possono essere disabilitate impostando **`.spec.metrics.enabled`** su `false`.

Prometheus è un database di serie temporali e un motore di valutazione delle regole per metriche. I contenitori IBM MQ espongono un endpoint di metrica che può essere interrogato da Prometheus. Le metriche vengono generate dagli argomenti di sistema MQ per il monitoraggio e la traccia dell'attività.

Red Hat OpenShift Container Platform include uno stack di monitoraggio preconfigurato, preinstallato e con aggiornamento automatico che utilizza un server Prometheus . Lo stack di controllo Red Hat OpenShift Container Platform deve essere configurato per monitorare i progetti definiti dall'utente. Per ulteriori informazioni, consultare Abilitazione del monitoraggio per i progetti definiti dall'utente. IBM MQ Operator crea un `ServiceMonitor` quando crei un `QueueManager` con le metriche abilitate, che l'operatore Prometheus può quindi rilevare.

Nelle versioni precedenti di IBM Cloud Pak for Integration, potresti anche utilizzare il servizio IBM Cloud Platform Monitoring per fornire invece un server Prometheus .

OpenShift CP4I *Metriche pubblicate quando si utilizza IBM MQ Operator*

I contenitori dei gestori code possono pubblicare metriche compatibili con Red Hat OpenShift Monitoring.

Metrica	Tipo	Descrizione
`ibmmq_qmgr_commit_total`	`counter`	Conteggio commit
`ibmmq_qmgr_cpu_load_fifteen_minute_average_percentage`	`gauge`	Carico CPU - media di quindici minuti
`ibmmq_qmgr_cpu_load_five_minute_average_percentage`	`gauge`	Carico CPU - media di cinque minuti
`ibmmq_qmgr_cpu_load_one_minute_average_percentage`	`gauge`	Carico CPU - media di un minuto
`ibmmq_qmgr_destructive_get_bytes_total`	`counter`	Totale estrazioni distruttive dell'intervallo - conteggio byte
`ibmmq_qmgr_destructive_get_total`	`counter`	Totale estrazioni distruttive dell'intervallo - conteggio
`ibmmq_qmgr_durable_subscription_alter_total`	`counter`	Conteggio modifiche sottoscrizioni durature

Metrica	Tipo	Descrizione
ibmmq_qmgr_durable_subscription_create_total	counter	Conteggio creazioni sottoscrizioni durature
ibmmq_qmgr_durable_subscription_delete_total	counter	Conteggio eliminazione sottoscrizioni durature
ibmmq_qmgr_durable_subscription_resume_total	counter	Conteggio ripristini sottoscrizioni durature
ibmmq_qmgr_errors_file_system_free_space_percentage	gauge	File system errori MQ - spazio disponibile
ibmmq_qmgr_errors_file_system_in_use_bytes	gauge	File system errori MQ - byte in uso
ibmmq_qmgr_expired_message_total	counter	Conteggio messaggi scaduti
ibmmq_qmgr_failed_browse_total	counter	Conteggio esplorazioni non riuscite
ibmmq_qmgr_failed_mqcb_total	counter	Conteggio MQCB non riusciti
ibmmq_qmgr_failed_mqclose_total	counter	Conteggio MQCLOSE non riusciti
ibmmq_qmgr_failed_mqconn_mqconnx_total	counter	Conteggio MQCONN/MQCONNX non riusciti
ibmmq_qmgr_failed_mqget_total	counter	MQGET non riusciti - conteggio
ibmmq_qmgr_failed_mqinq_total	counter	Conteggio MQINQ non riusciti
ibmmq_qmgr_failed_mqopen_total	counter	Conteggio MQOPEN non riusciti
ibmmq_qmgr_failed_mqput1_total	counter	Conteggio MQPUT1 non riusciti
ibmmq_qmgr_failed_mqput_total	counter	Conteggio MQPUT non riusciti
ibmmq_qmgr_failed_mqset_total	counter	Conteggio MQSET non riusciti
ibmmq_qmgr_failed_mqsubrq_total	counter	Conteggio MQSUBRQ non riusciti
ibmmq_qmgr_failed_subscription_create_alter_resume_total	counter	Conteggio creazioni/modifiche/ripristini sottoscrizioni non riusciti

Metrica	Tipo	Descrizione
ibmmq_qmgr_failed_subscription_delete_total	counter	Conteggio errore di eliminazione sottoscrizione
ibmmq_qmgr_failed_topic_mqput_mqput1_total	counter	Conteggio MQPUT/MQPUT1 di argomento non riusciti
ibmmq_qmgr_fdc_files	gauge	Conteggio file FDC MQ
ibmmq_qmgr_log_file_system_in_use_bytes	gauge	File system log - byte in uso
ibmmq_qmgr_log_file_system_max_bytes	gauge	File system log - massimo di byte
ibmmq_qmgr_log_in_use_bytes	gauge	Log - byte in uso
ibmmq_qmgr_log_logical_written_bytes_total	counter	Log - byte logici scritti
ibmmq_qmgr_log_max_bytes	gauge	Log - massimo di byte
ibmmq_qmgr_log_physical_written_bytes_total	counter	Log - byte fisici scritti
ibmmq_qmgr_log_primary_space_in_use_percentage	gauge	Log - spazio primario corrente in uso
ibmmq_qmgr_log_workload_primary_space_utilization_percentage	gauge	Log - utilizzo spazio primario carico di lavoro
ibmmq_qmgr_log_write_latency_seconds	gauge	Log - latenza scrittura
ibmmq_qmgr_log_write_size_bytes	gauge	Log - dimensione scrittura
ibmmq_qmgr_mqcb_total	counter	Conteggio MQCB
ibmmq_qmgr_mqclose_total	counter	Conteggio MQCLOSE
ibmmq_qmgr_mqconn_mqconnx_total	counter	Conteggio MQCONN/MQCONNX
ibmmq_qmgr_mqctl_total	counter	Conteggio MQCTL
ibmmq_qmgr_mqdisc_total	counter	Conteggio MQDISC

Metrica	Tipo	Descrizione
ibmmq_qmgr_mqinq_total	counter	Conteggio MQINQ
ibmmq_qmgr_mqopen_total	counter	Conteggio MQOPEN
ibmmq_qmgr_mqput_mqput1_bytes_total	counter	Conteggio byte totale MQPUT/MQPUT1 dell'intervallo
ibmmq_qmgr_mqput_mqput1_total	counter	Conteggio totale MQPUT/MQPUT1 dell'intervallo
ibmmq_qmgr_mqset_total	counter	Conteggio MQSET
ibmmq_qmgr_mqstat_total	counter	Conteggio MQSTAT
ibmmq_qmgr_mqsubrq_total	counter	Conteggio MQSUBRQ
ibmmq_qmgr_non_durable_subscription_create_total	counter	Conteggio creazioni sottoscrizioni non durature
ibmmq_qmgr_non_durable_subscription_delete_total	counter	Conteggio eliminazione sottoscrizioni non durature
ibmmq_qmgr_non_persistent_message_browse_bytes_total	counter	Esplorazione messaggi non permanenti - conteggio byte
ibmmq_qmgr_non_persistent_message_browse_total	counter	Esplorazione messaggi non permanenti - conteggio
ibmmq_qmgr_non_persistent_message_destructive_get_total	counter	Estrazione distruttiva di messaggi non permanenti - conteggio
ibmmq_qmgr_non_persistent_message_get_bytes_total	counter	Ottenimento messaggi non permanenti - conteggio byte
ibmmq_qmgr_non_persistent_message_mqput1_total	counter	Conteggio MQPUT1 messaggi non permanenti
ibmmq_qmgr_non_persistent_message_mqput_total	counter	Conteggio MQPUT messaggi non permanenti
ibmmq_qmgr_non_persistent_message_put_bytes_total	counter	Inserimento messaggi non permanenti - conteggio byte
ibmmq_qmgr_non_persistent_topic_mqput_mqput1_total	counter	Non permanente - conteggio MQPUT/MQPUT1 di argomento

Metrica	Tipo	Descrizione
ibmmq_qmgr_persistent_message_browse_bytes_total	counter	Esplorazione messaggi permanenti - conteggio byte
ibmmq_qmgr_persistent_message_browse_total	counter	Esplorazione messaggi permanenti - conteggio
ibmmq_qmgr_persistent_message_destructive_get_total	counter	Estrazione distruttiva di messaggi permanenti - conteggio
ibmmq_qmgr_persistent_message_get_bytes_total	counter	Ottenimento messaggi permanenti - conteggio byte
ibmmq_qmgr_persistent_message_mqput1_total	counter	Conteggio MQPUT1 messaggi permanenti
ibmmq_qmgr_persistent_message_mqput_total	counter	Conteggio MQPUT messaggi permanenti
ibmmq_qmgr_persistent_message_put_bytes_total	counter	Inserimento messaggi permanenti - conteggio byte
ibmmq_qmgr_persistent_topic_mqput_mqput1_total	counter	Permanente - conteggio MQPUT/MQPUT1 di argomento
ibmmq_qmgr_published_to_subscribers_bytes_total	counter	Pubblicato per i sottoscrittori - conteggio byte
ibmmq_qmgr_published_to_subscribers_message_total	counter	Pubblicato per i sottoscrittori - conteggio messaggi
ibmmq_qmgr_purged_queue_total	counter	Conteggio code eliminate
ibmmq_qmgr_queue_manager_file_system_free_space_percentage	gauge	File system Gestore code - spazio disponibile
ibmmq_qmgr_queue_manager_file_system_in_use_bytes	gauge	File system Gestore code - byte in uso
ibmmq_qmgr_ram_free_percentage	gauge	Percentuale RAM disponibile
ibmmq_qmgr_ram_usage_estimate_for_queue_manager_bytes	gauge	Totale byte della RAM - stima per il gestore code
ibmmq_qmgr_rollback_total	counter	Conteggio rollback

Metrica	Tipo	Descrizione
ibmmq_qmgr_system_cpu_time_estimate_for_queue_manager_percentage	gauge	Tempo CPU di sistema - stima percentuale per il gestore code
ibmmq_qmgr_system_cpu_time_percentage	gauge	Percentuale di tempo CPU di sistema
ibmmq_qmgr_topic_mput_mput1_total	counter	Totale dell'intervallo di MQPUT/MQPUT1 di argomenti
ibmmq_qmgr_topic_put_bytes_total	counter	Inserimento totale byte di argomento dell'intervallo
ibmmq_qmgr_trace_file_system_free_space_percentage	gauge	File system traccia MQ - spazio disponibile
ibmmq_qmgr_trace_file_system_in_use_bytes	gauge	File system traccia MQ - byte in uso
ibmmq_qmgr_user_cpu_time_estimate_for_queue_manager_percentage	gauge	Tempo CPU utente - stima percentuale per il gestore code
ibmmq_qmgr_user_cpu_time_percentage	gauge	Percentuale di tempo CPU utente

CP4I V9.2.2 CD Visualizzazione dello stato dei gestori code della HA nativa per i contenitori certificati IBM MQ

Per i contenitori certificati IBM MQ , puoi visualizzare lo stato delle istanze Native HA eseguendo il comando **dspmq** all'interno di uno dei pod in esecuzione.

Informazioni su questa attività

Importante:

È possibile utilizzare il comando **dspmq** in uno dei pod in esecuzione per visualizzare lo stato operativo di un'istanza del gestore code. Le informazioni restituite dipendono dal fatto che l'istanza sia attiva o una replica. Le informazioni fornite dall'istanza attiva sono definitive, le informazioni dai nodi di replica potrebbero non essere aggiornate.

È possibile effettuare le seguenti azioni:

- Visualizzare se l'istanza del gestore code sul nodo corrente è attiva o una replica.
- Visualizza lo stato operativo della HA nativa dell'istanza sul nodo corrente.
- Visualizzare lo stato operativo di tutte e tre le istanze in una configurazione HA nativa.

I seguenti campi di stato vengono utilizzati per riportare lo stato di configurazione della HA nativa:

Ruolo
Specifica il ruolo corrente dell'istanza ed è uno tra `Active`, `Replica`o `Unknown`.

ISTANZA
Il nome fornito per questa istanza del gestore code quando è stata creata utilizzando l'opzione **-lr** del comando **crtmqm** .

INSYNC
Indica se l'istanza è in grado di assumere il controllo come istanza attiva, se richiesto.

Quorum
Riporta lo stato del quorum nel formato *number_of_instances_in - sync/ number_of_instances_configured*.

REPLADDR
L'indirizzo di replica dell'istanza del gestore code.

COLLEGA
Indica se il nodo è connesso all'istanza attiva.

BACKLOG
Indica il numero di KB in cui si trova l'istanza.

CONNETTIN
Indica se l'istanza denominata è connessa a questa istanza.

ALTDATE
Indica la data in cui queste informazioni sono state aggiornate l'ultima volta (vuoto se non sono mai state aggiornate).

ALTTIME
Indica l'ora in cui queste informazioni sono state aggiornate l'ultima volta (vuoto se non sono mai state aggiornate).

Procedura

- Trova i pod che fanno parte del tuo gestore code.

```
oc get pod --selector app.kubernetes.io/instance=nativeha-qm
```

- Esegui il dspmq in uno dei pod

```
oc exec -t Pod dspmq
```

```
oc rsh Pod
```

per una shell interattiva, dove è possibile eseguire direttamente dspmq .

- Per determinare se un'istanza del gestore code è in esecuzione come istanza attiva o come replica:

```
oc exec -t Pod dspmq -o status -m QMgrName
```

Un'istanza attiva di un gestore code denominato BOB riporta il seguente stato:

```
QMNAME(BOB)                STATUS(Running)
```

Un'istanza di replica di un gestore code denominato BOB riporta il seguente stato:

```
QMNAME(BOB)                STATUS(Replica)
```

Un'istanza inattiva riporta il seguente stato:

```
QMNAME(BOB)                STATUS(Ended Immediately)
```

- Per determinare lo stato operativo della HA nativa dell'istanza nel pod specificato:

```
oc exec -t Pod dspmq -o nativeha -m QMgrName
```

L'istanza attiva di un gestore code denominato BOB potrebbe riportare il seguente stato:

```
QMNAME(BOB)               ROLE(Active) INSTANCE(inst1) INSYNC(Yes) QUORUM(3/3)
```

Un'istanza di replica di un gestore code denominato BOB potrebbe riportare il seguente stato:

```
QMNAME(BOB)               ROLE(Replica) INSTANCE(inst2) INSYNC(Yes) QUORUM(2/3)
```

Un'istanza inattiva di un gestore code denominato BOB potrebbe riportare il seguente stato:

```
QMNAME(BOB)               ROLE(Unknown) INSTANCE(inst3) INSYNC(no) QUORUM(0/3)
```

- Per determinare lo stato operativo della HA nativo di tutte le istanze nella configurazione della HA nativa:

```
oc exec -t Pod dspmq -o nativeha -x -m QMgrName
```

Se si immette questo comando sul nodo che esegue l'istanza attiva del BOB del gestore code, è possibile che si riceva il seguente stato:

```
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)
```

Se si immette questo comando su un nodo che esegue un'istanza di replica del BOB del gestore code, è possibile che si riceva il seguente stato, che indica che una delle repliche è in ritardo:

```
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)
```

Se si immette questo comando su un nodo che esegue un'istanza inattiva del BOB del gestore code, è possibile che si riceva il seguente stato:

```
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()
```

Se si immette il comando quando le istanze stanno ancora negoziando quali sono attive e quali sono repliche, si riceverà il seguente stato:

```
QMNAME(BOB)              STATUS(Negotiating)
```

Riferimenti correlati

comando dspmq (visualizza gestori code)

“Esempio: configurazione di un gestore code HA nativo” a pagina 94

Questo esempio mostra come distribuire un gestore code utilizzando la funzione alta disponibilità nativa in Red Hat OpenShift Container Platform (OCP) utilizzando IBM MQ Operator.

OpenShift CP4I Backup e ripristino della configurazione del gestore code utilizzando la CLI Red Hat OpenShift

Il backup della configurazione del gestore code può essere utile per ricreare un gestore code dalle relative definizioni se la configurazione del gestore code viene persa. Questa procedura non esegue il backup dei dati di log del gestore code. A causa della natura transitoria dei messaggi, i dati di log cronologici sono probabilmente irrilevanti al momento del ripristino.

Prima di iniziare

Accedi al tuo cluster utilizzando **`cloudctl login`** (per IBM Cloud Pak for Integration) o **`oc login`**.

Procedura

- Eseguire il back up della configurazione del gestore code.

  È possibile utilizzare il comando **`dmpmqcfg`** per eseguire il dump della configurazione di un gestore code IBM MQ .

  a) Ottenere il nome del pod per il gestore code.
  Ad esempio, è possibile eseguire il seguente comando, dove *queue_manager_name* è il nome della risorsa `QueueManager` :

  ```
  oc get pods --selector app.kubernetes.io/name=ibm-mq,app.kubernetes.io/
  instance=queue_manager_name
  ```

  b) Eseguire il comando **`dmpmqcfg`** sul pod, indirizzando l'emissione in un file sulla macchina locale.

  `dmpmqcfg` emette la configurazione MQSC del gestore code.

  ```
  oc exec -it pod_name -- dmpmqcfg > backup.mqsc
  ```

- Ripristinare la configurazione del gestore code.

  Dopo aver seguito la procedura di backup descritta nel passo precedente, si dovrebbe avere un file `backup.mqsc` che contiene la configurazione del gestore code. È possibile ripristinare la configurazione applicando questo file a un nuovo gestore code.

  a) Ottenere il nome del pod per il gestore code.
  Ad esempio, è possibile eseguire il seguente comando, dove *queue_manager_name* è il nome della risorsa `QueueManager` :

  ```
  oc get pods --selector app.kubernetes.io/name=ibm-mq,app.kubernetes.io/
  instance=queue_manager_name
  ```

  b) Eseguire il comando **`runmqsc`** sul pod, indirizzando il contenuto del file `backup.mqsc` .

  ```
  oc exec -i pod_name -- runmqsc < backup.mqsc
  ```

OpenShift CP4I Risoluzione dei problemi con IBM MQ Operator

Se si stanno riscontrando dei problemi con IBM MQ Operator, utilizzare le tecniche descritte per facilitarne la diagnostica e la soluzione.

Procedura

- “Risoluzione dei problemi: accesso ai dati del gestore code” a pagina 124

OpenShift CP4I Risoluzione dei problemi: accesso ai dati del gestore code

Utilizza lo strumento di controllo PVC per ottenere l'accesso ai file su una PVC del gestore code in cui non può essere stabilita una shell remota per il pod del gestore code. Ciò potrebbe essere dovuto al fatto che il pod è in uno stato **`Error`** o **`CrashLoopBackOff`** . Questo strumento è progettato per essere utilizzato con i gestori code distribuiti da IBM MQ Operator.

Prima di iniziare

Per utilizzare lo strumento di controllo PVC. è necessario avere accesso al proprio spazio nomi del gestore code.

Informazioni su questa attività

Per risolvere i problemi, è possibile accedere ai dati memorizzati nelle PVC (Persistent Volume Claims) associate a un determinato gestore code. Per fare ciò, utilizza uno strumento per montare le PVC in un insieme di pod inspector. Puoi quindi ottenere una shell remota in qualsiasi pod inspector per leggere i file.

A seconda del tipo di distribuzione, vengono creati tra uno e tre pod inspector. I volumi specifici per un determinato pod di un gestore code Native - HA o Multi - Instance sono disponibili sul pod inspector PVC associato. I volumi condivisi sono disponibili su tutti gli ispettori. Il nome del pod inspector contiene il nome del pod del gestore code associato.

Procedura

1. Scarica lo strumento MQ PVC inspector.

   Lo strumento è disponibile qui: https://github.com/ibm-messaging/mq-pvc-tool.
2. Assicurati di essere collegato al cluster.
3. Individuare il nome del gestore code e lo spazio dei nomi in cui è in esecuzione il gestore code.
4. Eseguire lo strumento inspector sul tuo gestore code.
   a) Eseguire il seguente comando, specificando il nome del gestore code e il relativo nome spazio dei nomi.

   ```
   ./pvc-tool.sh queue_manager_name queue_manager_namespace_name
   ```

   b) Una volta completato lo strumento, immetti il seguente comando per visualizzare i pod inspector in fase di creazione.

   ```
   oc get pods
   ```

5. Visualizza i file montati nel pod inspector.
   a) Ogni pod di controllo PVC è associato a un pod del gestore code, quindi potrebbero esserci più pod di controllo. Accedi a uno di questi pod, immettendo il seguente comando:

   ```
   oc rsh pvc-inspector-pod-name
   ```

   L'utente viene collocato nella directory contenente le directory PVC montate.
   b) Apri una shell remota nel pod, immettendo il seguente comando:

   ```
   ls
   ```

   c) È possibile visualizzare le directory con lo stesso nome delle PVC che sono state montate. Accedere ai file sulle PVC del gestore code sfogliando queste directory. Per visualizzare un elenco di PVC, eseguire il seguente comando al di fuori della sessione della shell remota:

   ```
   oc get pvc
   ```

   d) Ripulisci i pods creati dallo strumento, immettendo il seguente comando:

   ```
   'oc delete pods -l tool=mq-pvc-inspector
   ```

OpenShift CP4I Riferimento API per IBM MQ Operator

IBM MQ fornisce un operatore Kubernetes , che fornisce un'integrazione nativa con Red Hat OpenShift Container Platform.

OpenShift CP4I Riferimento API per mq.ibm.com/v1beta1

L'API v1beta1 può essere utilizzata per creare e gestire risorse QueueManager .

Riferimento per la licenza per mq.ibm.com/v1beta1

Versioni di licenza correnti

Il campo `spec.license.license` deve contenere l'identificativo della licenza che si sta accettando. I valori validi sono:

Valore di `spec.license.license`	Valore di `spec.license.use`	Informazioni sulla licenza	Versioni IBM MQ applicabili
`L-RJON-C7QG3S`	`Production` o `NonProduction`	IBM Cloud Pak for Integration 2021.4.1	9.2.4 o 9.2.5
`L-RJON-C7QFZX`	`Production` o `NonProduction`	IBM Cloud Pak for Integration Edizione limitata 2021.4.1	9.2.4 o 9.2.5
`L-RJON-C5CSNH`	`Production` o `NonProduction`	IBM Cloud Pak for Integration 2021.3.1	9.2.3 o 9.2.4
`L-RJON-C5CSM2`	`Production` o `NonProduction`	IBM Cloud Pak for Integration Edizione limitata 2021.3.1	9.2.3 o 9.2.4
`L-RJON-BZFQU2`	`Production` o `NonProduction`	IBM Cloud Pak for Integration 2021.2.1	9.2.3
`L-RJON-BZFQSB`	`Production` o `NonProduction`	IBM Cloud Pak for Integration Edizione limitata 2021.2.1	9.2.3
`L-RJON-BUVMQX`	`Production` o `NonProduction`	IBM Cloud Pak for Integration 2020.4.1	9.2.0 EUS o 9.2.1
`L-RJON-BUVMYB`	`Production` o `NonProduction`	IBM Cloud Pak for Integration Edizione limitata 2020.4.1	9.2.0 EUS o 9.2.1
`L-APIG-BZDDDY`	`Production`	IBM MQ Advanced e IBM MQ Advanced per ambiente non di produzione 9.2 - 07/2021	9.2.3, 9.2.4 o 9.2.5
`L-APIG-BYHCL7`	`Development`	IBM MQ Advanced for Developers (Non Warranted) V9.2 - 07/2021	9.2.3, 9.2.4 o 9.2.5
`L-APIG-BVJJB3`	`Production`	IBM MQ Advanced e IBM MQ Advanced per ambienti non di produzione 9.2 - 03/2021	9.2.2
`L-APIG-BMJJBM`	`Production`	IBM MQ Advanced V9.2	9.2.0 CD o 9.2.1
`L-APIG-BMKG5H`	`Development`	IBM MQ Advanced for Developers (Non Warranted) V9.2	CD 9.2.0 , 9.2.1 o 9.2.2

Notare che la licenza *versione* è specificata, che non è sempre uguale alla versione di IBM MQ.

Versioni di licenza precedenti

Il campo `spec.license.license` deve contenere l'identificativo della licenza che si sta accettando. I valori validi sono:

Valore di **spec.license.license**	Valore di **spec.license.use**	Informazioni sulla licenza	Versioni IBM MQ applicabili
L-RJON-BXUPZ2	Production o NonProduction	IBM Cloud Pak for Integration 2021.1.1	9.2.2
L-RJON-BXUQ34	Production o NonProduction	IBM Cloud Pak for Integration Edizione limitata 2021.1.1	9.2.2
L-RJON-BYRMYW	NonProduction	IBM Cloud Pak for Integration Eval - Demo 2021.1.1. Release iniziale da utilizzare solo con Native HA con IBM MQ Operator 1.5 .	9.2.2
L-RJON-BQPGWD	Production o NonProduction	IBM Cloud Pak for Integration 2020.3.1	CD 9.2.0
L-RJON-BN7PN3	Production o NonProduction	IBM Cloud Pak for Integration 2020.2.1	CD 9.1.5 o 9.2.0
L-RJON-BPHL2Y	Production o NonProduction	IBM Cloud Pak for Integration Edizione limitata 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 Warranted) V9.1 - 04/2020	9.1.5

Notare che la licenza *versione* è specificata, che non è sempre uguale alla versione di IBM MQ.

OpenShift CP4I ***Riferimento API per QueueManager (mq.ibm.com/v1beta1)***

QueueManager

Un QueueManager è un IBM MQ che fornisce servizi di accodamento e pubblicazione / sottoscrizione alle applicazioni.

Campo	Descrizione
apiVersion Stringa	APIVersion definisce lo schema con versione di questa rappresentazione di oggetto. I server devono convertire gli schemi riconosciuti nell'ultimo valore interno e possono rifiutare i valori non riconosciuti. Ulteriori informazioni: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources.
kind Stringa	Il tipo è un valore stringa che rappresenta la risorsa REST rappresentata da questo oggetto. I server possono dedurre questo dall'endpoint a cui il client inoltra le richieste. Non può essere aggiornato. In CamelCase. Ulteriori informazioni: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-tipi.
metadata	
spec QueueManagerSpec	Lo stato desiderato del QueueManager.
status QueueManagergestore code	Lo stato osservato di QueueManager.

.spec

Lo stato desiderato del QueueManager.

Viene visualizzato in:

- "QueueManager" a pagina 127

Campo	Descrizione
`affinity`	Regole di affinità Kubernetes standard. Per ulteriori informazioni, consultare https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.17/#affinity-v1-core.
`annotations` Annotazioni	Il campo delle annotazioni funge da pass-through per le annotazioni Pod. Gli utenti possono aggiungere qualsiasi annotazione a questo campo e applicarlo al pod. Le annotazioni qui sovrascrivono le annotazioni predefinite, se fornite. Richiede l'operatore MQ 1.3.0 o superiore.
Array `imagePullSecrets` LocalObjectReference	Un elenco facoltativo di riferimenti ai segreti nello stesso spazio dei nomi da utilizzare per il pull delle immagini utilizzate da questo QueueManager. Se specificato, questi segreti verranno passati alle singole implementazioni del programma di estrattore per utilizzarli. Ad esempio, nel caso di docker, vengono rispettati solo i segreti di tipo DockerConfig . Per ulteriori informazioni, vedi https://kubernetes.io/docs/concepts/containers/images#specifying- imagepullsecrets - on - a - pod.
`labels` Etichette	Il campo Etichette funge da pass-through per le etichette Pod. Gli utenti possono aggiungere qualsiasi etichetta a questo campo e applicarlo al pod. Le etichette qui sovrascrivono le etichette predefinite, se fornite. Richiede l'operatore MQ 1.3.0 o superiore.
`license` Licenza	Impostazioni che controllano l'accettazione della licenza e quali metriche di licenza utilizzare.
`pki` PKI	Impostazioni Public Key Infrastructure, per la definizione di chiavi e certificati da utilizzare con TLS (Transport Layer Security) o AMS (Advanced Message Security) MQ Advanced Message Security .
`queueManager` Configurazione QueueManager	Impostazioni per il contenitore Gestore code e il gestore code sottostante.
`securityContext` SecurityContext	Impostazioni di protezione da aggiungere al securityContextdi Queue Manager Pod.
`template` Modello	Templating avanzato per risorse Kubernetes . Il template consente agli utenti di sovrascrivere il modo in cui IBM MQ genera le risorse Kubernetes sottostanti, come StatefulSet, Pods e Services. Questo è solo per gli utenti avanzati, poiché ha il potenziale di interrompere il normale funzionamento di MQ se utilizzato in modo non corretto. Tutti i valori specificati altrove nella risorsa QueueManager verranno sovrascritti dalle impostazioni nel modello.
`terminationGracePeriodSeconds` intero	Durata facoltativa in secondi di cui il pod ha bisogno per terminare correttamente. Il valore deve essere un numero intero non negativo. Il valore zero indica l'eliminazione immediata. L'ora di destinazione in cui si tenta di terminare il gestore code, eseguendo l'escalation delle fasi di disconnessione dell'applicazione. Le attività essenziali di manutenzione del gestore code vengono interrotte, se necessario. Il valore predefinito è 30 secondi.
`tracing` TracingConfig	Impostazioni per l'integrazione di traccia con il dashboard Operazioni Cloud Pak for Integration .

Campo	Descrizione
`version` Stringa	Impostazione che controlla la versione di MQ che verrà utilizzata (obbligatorio). Ad esempio: `9.1.5.0-r2` specifica MQ versione 9.1.5.0, utilizzando la seconda revisione dell'immagine contenitore. Le correzioni specifiche del contenitore vengono spesso applicate nelle revisioni, come le correzioni all'immagine di base.
`web` WebServerConfigurazione	Impostazioni per il server web MQ .

.spec.annotations

Il campo delle annotazioni funge da pass-through per le annotazioni Pod. Gli utenti possono aggiungere qualsiasi annotazione a questo campo e applicarlo al pod. Le annotazioni qui sovrascrivono le annotazioni predefinite, se fornite. Richiede l'operatore MQ 1.3.0 o superiore.

Viene visualizzato in:

- ".spec" a pagina 127

.spec.imagePullSecrets

LocalObjectIl riferimento contiene informazioni sufficienti per individuare l'oggetto di riferimento all'interno dello stesso spazio dei nomi.

Viene visualizzato in:

- ".spec" a pagina 127

Campo	Descrizione
`name` Stringa	Nome del referente. Ulteriori informazioni: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names TODO: aggiungere altri campi utili. apiVersion, tipo, uid?.

.spec.labels

Il campo Etichette funge da pass-through per le etichette Pod. Gli utenti possono aggiungere qualsiasi etichetta a questo campo e applicarlo al pod. Le etichette qui sovrascrivono le etichette predefinite, se fornite. Richiede l'operatore MQ 1.3.0 o superiore.

Viene visualizzato in:

- ".spec" a pagina 127

.spec.license

Impostazioni che controllano l'accettazione della licenza e quali metriche di licenza utilizzare.

Viene visualizzato in:

- ".spec" a pagina 127

Campo	Descrizione
`accept` booleano	Indica se si accetta o meno la licenza associata a questo software (obbligatorio).
`license` Stringa	L'identificativo della licenza che si sta accettando. Deve essere l'identificativo di licenza corretto per la versione di MQ che si utilizza. Consultare http://ibm.biz/BdqvCF per i valori validi.

Campo	Descrizione
`metric` Stringa	Impostazione che specifica quale metrica di licenza utilizzare. Ad esempio, `ProcessorValueUnit`, `VirtualProcessorCore` o `ManagedVirtualServer`. Il valore predefinito è `ProcessorValueUnit` quando si usa una licenza di MQ e `VirtualProcessorCore` quando si utilizza una licenza di Cloud Pak for Integration .
`use` Stringa	Impostazione che controlla il modo in cui verrà utilizzato il software, dove la licenza supporta più utilizzi. Consultare http://ibm.biz/BdqvCF per i valori validi.

.spec.pki

Impostazioni Public Key Infrastructure, per la definizione di chiavi e certificati da utilizzare con TLS (Transport Layer Security) o AMS (Advanced Message Security) MQ Advanced Message Security .

Viene visualizzato in:

- ".spec" a pagina 127

Campo	Descrizione
Array `keys` PKISource	Chiavi private da aggiungere al repository delle chiavi del gestore code.
Array `trust` PKISource	Certificati da aggiungere al repository delle chiavi del gestore code.

.spec.pki.keys

PKISource definisce un'origine delle informazioni Public Key Infrastructure, come chiavi o certificati.

Viene visualizzato in:

- ".spec.pki" a pagina 130

Campo	Descrizione
`name` Stringa	Il nome viene utilizzato come etichetta per la chiave o il certificato. Deve essere una stringa alfanumerica minuscola.
`secret` Segreto	Fornisci una chiave utilizzando un segreto Kubernetes .

.spec.pki.keys.secret

Fornisci una chiave utilizzando un segreto Kubernetes .

Viene visualizzato in:

- ".spec.pki.keys" a pagina 130

Campo	Descrizione
`items` schiera	Le chiavi all'interno del segreto Kubernetes che devono essere aggiunte al contenitore Gestore code.
`secretName` Stringa	Il nome del segreto Kubernetes .

.spec.pki.trust

PKISource definisce un'origine delle informazioni Public Key Infrastructure, come chiavi o certificati.

Viene visualizzato in:

- ".spec.pki" a pagina 130

Campo	Descrizione
name Stringa	Il nome viene utilizzato come etichetta per la chiave o il certificato. Deve essere una stringa alfanumerica minuscola.
secret Segreto	Fornisci una chiave utilizzando un segreto Kubernetes .

.spec.pki.trust.secret

Fornisci una chiave utilizzando un segreto Kubernetes .

Viene visualizzato in:

- ".spec.pki.trust" a pagina 130

Campo	Descrizione
items schiera	Le chiavi all'interno del segreto Kubernetes che devono essere aggiunte al contenitore Gestore code.
secretName Stringa	Il nome del segreto Kubernetes .

.spec.queueManager

Impostazioni per il contenitore Gestore code e il gestore code sottostante.

Viene visualizzato in:

- ".spec" a pagina 127

Campo	Descrizione
availability Disponibilità	Impostazioni di disponibilità per il gestore code, ad esempio se utilizzare o meno una coppia active - standby o un'alta disponibilità nativa.
debug booleano	Indica se registrare o meno i messaggi di debug dal codice specifico del contenitore al log del contenitore. Il valore predefinito è false.
image Stringa	L'immagine contenitore che verrà utilizzata.
imagePullPolicy Stringa	Impostazione che controlla quando il kubelet tenta di estrarre l'immagine specificata. Il valore predefinito è IfNotPresent.
Array ini INISource	Impostazioni per fornire INI per il gestore code. Richiede l'operatore MQ 1.1.0 o superiore.
livenessProbe QueueManagerLivenessProbe	Impostazioni che controllano il probe di attività.
logFormat Stringa	Quale formato di log utilizzare per questo contenitore. Utilizza JSON per i log formattati JSON dal contenitore. Utilizzare Basic per i messaggi in formato testo. Il valore predefinito è Basic.
metrics QueueManagerMetriche	Impostazioni per le metriche in stile Prometheus.
Array mqsc MQSCSource	Impostazioni per fornire MQSC per il gestore code. Richiede l'operatore MQ 1.1.0 o superiore.
name Stringa	Nome del gestore code MQ sottostante, se diverso da metadata.name. Utilizzare questo campo se si desidera un nome gestore code non conforme alle regole Kubernetes per i nomi (ad esempio, un nome che include lettere maiuscole).

Campo	Descrizione
`readinessProbe` QueueManagerReadinessProbe	Impostazioni che controllano il probe di disponibilità.
`resources` Risorse	Impostazioni che controllano i requisiti delle risorse.
`route` Instrada	Impostazioni per l'instradamento del gestore code. Richiede l'operatore MQ 1.4.0 o superiore.
`startupProbe` StartupProbe	Impostazioni che controllano il probe di avvio. Si applica solo alle distribuzioni MultiInstance e NativeHA . Richiede l'operatore MQ 1.5.0 o superiore.
`storage` QueueManagerStorage	Impostazioni di archiviazione per controllare l'utilizzo da parte del gestore code di volumi persistenti e classi di archiviazione.

.spec.queueManager.availability

Impostazioni di disponibilità per il gestore code, ad esempio se utilizzare o meno una coppia active - standby o un'alta disponibilità nativa.

Viene visualizzato in:

- ".spec.queueManager" a pagina 131

Campo	Descrizione
`tls` TL	Impostazioni TLS facoltative per la configurazione della comunicazione protetta tra le repliche NativeHA . Richiede l'operatore MQ 1.5.0 o superiore.
`type` Stringa	Il tipo di disponibilità da utilizzare. Utilizza `SingleInstance` per un singolo pod, che verrà riavviato automaticamente (in alcuni casi) da Kubernetes. Utilizzare `MultiInstance` per una coppia di pod, uno dei quali è il gestore code `active` e l'altro è uno standby. Utilizzare `NativeHA` per la replica nativa ad alta disponibilità (richiede MQ Operator 1.5.0 o superiore). Il valore predefinito è `SingleInstance`. Per ulteriori dettagli, consultare http://ibm.biz/BdqAQa .
`updateStrategy` Stringa	La strategia di aggiornamento da utilizzare per i gestori code MultiInstance e NativeHA . Utilizzare `RollingUpdate` per abilitare gli aggiornamenti a rotazione automatica ogni volta che viene modificata la configurazione del gestore code. Utilizzare `OnDelete` per disabilitare gli aggiornamenti a rotazione automatica, le modifiche del gestore code verranno applicate solo quando i pod vengono eliminati (incluse le eliminazioni di pod attivate da fattori esterni). Il valore predefinito è `RollingUpdate`. Richiede l'operatore MQ 1.6.0 o superiore.

.spec.queueManager.availability.tls

Impostazioni TLS facoltative per la configurazione della comunicazione protetta tra le repliche NativeHA . Richiede l'operatore MQ 1.5.0 o superiore.

Viene visualizzato in:

- ".spec.queueManager.availability" a pagina 132

Campo	Descrizione
`cipherSpec` Stringa	Il nome di CipherSpec per TLS NativeHA .
`secretName` Stringa	Il nome del segreto Kubernetes .

.spec.queueManager.ini

Origine dei file di configurazione INI.

Viene visualizzato in:

- ".spec.queueManager" a pagina 131

Campo	Descrizione
`configMap` ConfigMapINISource	ConfigMap rappresenta una ConfigMap Kubernetes che contiene informazioni INI.
`secret` SecretINISource	Il segreto rappresenta un segreto Kubernetes che contiene informazioni INI.

.spec.queueManager.ini.configMap

ConfigMap rappresenta una ConfigMap Kubernetes che contiene informazioni INI.

Viene visualizzato in:

- ".spec.queueManager.ini" a pagina 133

Campo	Descrizione
`items` schiera	Le chiavi all'interno dell'origine Kubernetes che devono essere applicate.
`name` Stringa	Il nome dell'origine Kubernetes .

.spec.queueManager.ini.secret

Il segreto rappresenta un segreto Kubernetes che contiene informazioni INI.

Viene visualizzato in:

- ".spec.queueManager.ini" a pagina 133

Campo	Descrizione
`items` schiera	Le chiavi all'interno dell'origine Kubernetes che devono essere applicate.
`name` Stringa	Il nome dell'origine Kubernetes .

.spec.queueManager.livenessProbe

Impostazioni che controllano il probe di attività.

Viene visualizzato in:

- ".spec.queueManager" a pagina 131

Campo	Descrizione
`failureThreshold` intero	Numero minimo di errori consecutivi perché l'analisi venga considerata non riuscita dopo l'esito positivo. L'impostazione predefinita è 1.
`initialDelaySeconds` intero	Numero di secondi dopo l'avvio del contenitore prima dell'avvio dell'analisi. Il valore predefinito è 90 secondi per SingleInstance. Il valore predefinito è 0 secondi per le distribuzioni MultiInstance e NativeHA . Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
`periodSeconds` intero	La frequenza (in secondi) con cui eseguire l'analisi. Il valore predefinito è 10 secondi.

Campo	Descrizione
`successThreshold` intero	Numero minimo di esiti positivi consecutivi perché il probe venga considerato riuscito dopo l'errore. L'impostazione predefinita è 1.
`timeoutSeconds` intero	Numero di secondi dopo i quali l'analisi va in timeout. Il valore predefinito è 5 secondi. Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.queueManager.metrics

Impostazioni per le metriche in stile Prometheus.

Viene visualizzato in:

- ".spec.queueManager" a pagina 131

Campo	Descrizione
`enabled` booleano	Indica se abilitare o meno un endpoint per le metriche compatibili con Prometheus. L'impostazione predefinita è true.

.spec.queueManager.mqsc

Origine dei file di configurazione MQSC.

Viene visualizzato in:

- ".spec.queueManager" a pagina 131

Campo	Descrizione
`configMap` ConfigMapMQSCSource	ConfigMap rappresenta un ConfigMap Kubernetes che contiene informazioni MQSC.
`secret` SecretMQSCSource	Il segreto rappresenta un segreto Kubernetes che contiene informazioni MQSC.

.spec.queueManager.mqsc.configMap

ConfigMap rappresenta un ConfigMap Kubernetes che contiene informazioni MQSC.

Viene visualizzato in:

- ".spec.queueManager.mqsc" a pagina 134

Campo	Descrizione
`items` schiera	Le chiavi all'interno dell'origine Kubernetes che devono essere applicate.
`name` Stringa	Il nome dell'origine Kubernetes .

.spec.queueManager.mqsc.secret

Il segreto rappresenta un segreto Kubernetes che contiene informazioni MQSC.

Viene visualizzato in:

- ".spec.queueManager.mqsc" a pagina 134

Campo	Descrizione
`items` schiera	Le chiavi all'interno dell'origine Kubernetes che devono essere applicate.
`name` Stringa	Il nome dell'origine Kubernetes .

.spec.queueManager.readinessProbe

Impostazioni che controllano il probe di disponibilità.

Viene visualizzato in:

- ".spec.queueManager" a pagina 131

Campo	Descrizione
`failureThreshold` intero	Numero minimo di errori consecutivi perché l'analisi venga considerata non riuscita dopo l'esito positivo. L'impostazione predefinita è 1.
`initialDelaySeconds` intero	Numero di secondi dopo l'avvio del contenitore prima dell'avvio dell'analisi. Il valore predefinito è 10 secondi per SingleInstance. Il valore predefinito è 0 per le installazioni MultiInstance e NativeHA . Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
`periodSeconds` intero	La frequenza (in secondi) con cui eseguire l'analisi. Il valore predefinito è 5 secondi.
`successThreshold` intero	Numero minimo di esiti positivi consecutivi perché il probe venga considerato riuscito dopo l'errore. L'impostazione predefinita è 1.
`timeoutSeconds` intero	Numero di secondi dopo i quali l'analisi va in timeout. Il valore predefinito è 3 secondi. Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.queueManager.resources

Impostazioni che controllano i requisiti delle risorse.

Viene visualizzato in:

- ".spec.queueManager" a pagina 131

Campo	Descrizione
`limits` Limiti	Impostazioni CPU & memoria.
`requests` Richieste	Impostazioni CPU & memoria.

.spec.queueManager.resources.limits

Impostazioni CPU & memoria.

Viene visualizzato in:

- ".spec.queueManager.resources" a pagina 135

Campo	Descrizione
`cpu`	
`memory`	

.spec.queueManager.resources.requests

Impostazioni CPU & memoria.

Viene visualizzato in:

- ".spec.queueManager.resources" a pagina 135

Campo	Descrizione
`cpu`	
`memory`	

.spec.queueManager.route

Impostazioni per l'instradamento del gestore code. Richiede l'operatore MQ 1.4.0 o superiore.

Viene visualizzato in:

- ".spec.queueManager" a pagina 131

Campo	Descrizione
`enabled` booleano	Indica se abilitare o meno l'instradamento. L'impostazione predefinita è true.

.spec.queueManager.startupProbe

Impostazioni che controllano il probe di avvio. Si applica solo alle distribuzioni MultiInstance e NativeHA . Richiede l'operatore MQ 1.5.0 o superiore.

Viene visualizzato in:

- ".spec.queueManager" a pagina 131

Campo	Descrizione
`failureThreshold` intero	Numero minimo di errori consecutivi per il probe da considerare non riuscito. Il valore predefinito è 60.
`initialDelaySeconds` intero	Numero di secondi dopo l'avvio del contenitore prima dell'avvio dell'analisi. Il valore predefinito è 0 secondi. Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
`periodSeconds` intero	La frequenza (in secondi) con cui eseguire l'analisi. Il valore predefinito è 5 secondi.
`successThreshold` intero	Numero minimo di successi consecutivi per il probe da considerare riuscito. L'impostazione predefinita è 1.
`timeoutSeconds` intero	Numero di secondi dopo i quali l'analisi va in timeout. Il valore predefinito è 5 secondi. Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.queueManager.storage

Impostazioni di archiviazione per controllare l'utilizzo da parte del gestore code di volumi persistenti e classi di archiviazione.

Viene visualizzato in:

- ".spec.queueManager" a pagina 131

Campo	Descrizione
`defaultClass` Stringa	La classe di memoria da applicare a tutti i volumi permanenti di questo gestore code per impostazione predefinita. I volumi persistenti specifici possono definire la propria classe di archiviazione che sovrascriverà questa impostazione della classe di archiviazione predefinita. Se `type of availability` è `SingleInstance` o `NativeHA`, la classe di memoria può essere di tipo ReadWriteUna volta o ReadWriteMolti. Se `type of availability` è `MultiInstance`, la classe di memoria deve essere di tipo ReadWriteMany.
`defaultDeleteClaim` booleano	Indica se tutti i volumi devono essere eliminati o meno quando viene eliminato il gestore code. Volumi persistenti specifici possono definire il proprio valore per deleteClaim che sovrascriverà questa impostazione di richiesta defaultDelete. Il valore predefinito è false.
`persistedData` QueueManagerOptionalVolume	Dettagli PersistentVolume per i dati persistenti di MQ , inclusi la configurazione, code e messaggi. Obbligatorio quando si utilizza il gestore code a più istanze.
`queueManager` QueueManagerVolume	Il valore predefinito PersistentVolume per tutti i dati normalmente in `/var/mqm`. Conterrà tutti i dati persistenti e i log di recupero, se non vengono specificati altri volumi.
`recoveryLogs` QueueManagerOptionalVolume	Dettagli del volume persistente per i log di ripristino di MQ . Obbligatorio quando si utilizza il gestore code a più istanze.

.spec.queueManager.storage.persistedData

Dettagli PersistentVolume per i dati persistenti di MQ , inclusi la configurazione, code e messaggi. Obbligatorio quando si utilizza il gestore code a più istanze.

Viene visualizzato in:

- ".spec.queueManager.storage" a pagina 136

Campo	Descrizione
`class` Stringa	Classe di memoria da utilizzare per questo volume. Valido solo se `type` è `persistent-claim`. Se `type of availability` è `SingleInstance` o `NativeHA`, la classe di memoria può essere di tipo ReadWriteUna volta o ReadWriteMolti. Se `type of availability` è `MultiInstance`, la classe di memoria deve essere di tipo ReadWriteMany.
`deleteClaim` booleano	Indica se questo volume deve essere eliminato o meno quando viene eliminato il gestore code.
`enabled` booleano	Indica se questo volume deve essere abilitato o meno come volume separato o se deve essere posizionato sul volume `queueManager` predefinito. Il valore predefinito è false.
`size` Stringa	Dimensione del PersistentVolume da passare a Kubernetes, incluse le unità SI. Valido solo se `type` è `persistent-claim`. Ad esempio, `2Gi`. Il valore predefinito è `2Gi`.
`sizeLimit` Stringa	Limite dimensione quando si utilizza un volume `ephemeral` . I file sono ancora scritti in una directory temporanea, quindi è possibile utilizzare questa opzione per limitare la dimensione. Valido solo se `type` è `ephemeral`.

Campo	Descrizione
type Stringa	Tipo di volume da utilizzare. Scegli ephemeral per utilizzare l'archiviazione non persistente o persistent-claim per utilizzare un volume persistente. Il valore predefinito è persistent-claim.

.spec.queueManager.storage.queueManager

Il valore predefinito PersistentVolume per tutti i dati normalmente in /var/mqm. Conterrà tutti i dati persistenti e i log di recupero, se non vengono specificati altri volumi.

Viene visualizzato in:

- ".spec.queueManager.storage" a pagina 136

Campo	Descrizione
class Stringa	Classe di memoria da utilizzare per questo volume. Valido solo se type è persistent-claim. Se type of availability è SingleInstance o NativeHA, la classe di memoria può essere di tipo ReadWriteUna volta o ReadWriteMolti. Se type of availability è MultiInstance, la classe di memoria deve essere di tipo ReadWriteMany.
deleteClaim booleano	Indica se questo volume deve essere eliminato o meno quando viene eliminato il gestore code.
size Stringa	Dimensione del PersistentVolume da passare a Kubernetes, incluse le unità SI. Valido solo se type è persistent-claim. Ad esempio, 2Gi. Il valore predefinito è 2Gi.
sizeLimit Stringa	Limite dimensione quando si utilizza un volume ephemeral . I file sono ancora scritti in una directory temporanea, quindi è possibile utilizzare questa opzione per limitare la dimensione. Valido solo se type è ephemeral.
type Stringa	Tipo di volume da utilizzare. Scegli ephemeral per utilizzare l'archiviazione non persistente o persistent-claim per utilizzare un volume persistente. Il valore predefinito è persistent-claim.

.spec.queueManager.storage.recoveryLogs

Dettagli del volume persistente per i log di ripristino di MQ . Obbligatorio quando si utilizza il gestore code a più istanze.

Viene visualizzato in:

- ".spec.queueManager.storage" a pagina 136

Campo	Descrizione
class Stringa	Classe di memoria da utilizzare per questo volume. Valido solo se type è persistent-claim. Se type of availability è SingleInstance o NativeHA, la classe di memoria può essere di tipo ReadWriteUna volta o ReadWriteMolti. Se type of availability è MultiInstance, la classe di memoria deve essere di tipo ReadWriteMany.
deleteClaim booleano	Indica se questo volume deve essere eliminato o meno quando viene eliminato il gestore code.
enabled booleano	Indica se questo volume deve essere abilitato o meno come volume separato o se deve essere posizionato sul volume queueManager predefinito. Il valore predefinito è false.

Campo	Descrizione
`size` Stringa	Dimensione del PersistentVolume da passare a Kubernetes, incluse le unità SI. Valido solo se `type` è `persistent-claim`. Ad esempio, `2Gi`. Il valore predefinito è `2Gi`.
`sizeLimit` Stringa	Limite dimensione quando si utilizza un volume `ephemeral` . I file sono ancora scritti in una directory temporanea, quindi è possibile utilizzare questa opzione per limitare la dimensione. Valido solo se `type` è `ephemeral`.
`type` Stringa	Tipo di volume da utilizzare. Scegli `ephemeral` per utilizzare l'archiviazione non persistente o `persistent-claim` per utilizzare un volume persistente. Il valore predefinito è `persistent-claim`.

.spec.securityContext

Impostazioni di protezione da aggiungere al securityContextdi Queue Manager Pod.

Viene visualizzato in:

- ".spec" a pagina 127

Campo	Descrizione
`fsGroup` intero	Un gruppo supplementare speciale che si applica a tutti i contenitori in un pod. Alcuni tipi di volume consentono a Kubelet di modificare la proprietà di tale volume in modo che appartenga al pod: 1. Il GID proprietario sarà il FSGroup 2. Il bit setgid è impostato (i nuovi file creati nel volume saranno di proprietà di FSGroup) 3. I bit di autorizzazione sono OR ' d con rw-rw ---- Se non impostato, Kubelet non modificherà la proprietà e le autorizzazioni di alcun volume.
`initVolumeAsRoot` booleano	Ciò influenza il securityContext utilizzato dal contenitore che inizializza il PersistentVolume. Impostare questo valore su `true` se si sta utilizzando un provider di memoria che richiede di essere l'utente root per accedere ai volumi di cui è stato appena eseguito il provisioning. L'impostazione su `true` influisce sull'oggetto SCC (Security Context Constraints) che è possibile utilizzare e l'avvio del gestore code potrebbe non riuscire se non si è autorizzati ad utilizzare un SCC che consente l'utente root. Il valore predefinito è false. Per ulteriori informazioni, consultare https://docs.openshift.com/container-platform/latest/authentication/managing-security-context-constraints.html.
`supplementalGroups` schiera	Un elenco di gruppi applicati al primo processo eseguito in ogni contenitore, in aggiunta al GID primario del contenitore. Se non specificato, nessun gruppo verrà aggiunto ad alcun contenitore.

.spec.template

Templating avanzato per risorse Kubernetes . Il template consente agli utenti di sovrascrivere il modo in cui IBM MQ genera le risorse Kubernetes sottostanti, come StatefulSet, Pods e Services. Questo è solo per gli utenti avanzati, poiché ha il potenziale di interrompere il normale funzionamento di MQ se utilizzato in modo non corretto. Tutti i valori specificati altrove nella risorsa QueueManager verranno sovrascritti dalle impostazioni nel modello.

Viene visualizzato in:

- ".spec" a pagina 127

Campo	Descrizione
pod	Sovrascritture per il template utilizzato per il pod. Vedere https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.17/#podspec-v1-core.

.spec.tracing

Impostazioni per l'integrazione di traccia con il dashboard Operazioni Cloud Pak for Integration .

Viene visualizzato in:

- ".spec" a pagina 127

Campo	Descrizione
`agent` TracingAgent	Solo in Cloud Pak for Integration , è possibile configurare le impostazioni per l'agent di traccia facoltativo.
`collector` TracingCollector	Solo in Cloud Pak for Integration , è possibile configurare le impostazioni per il Tracing Collector facoltativo.
`enabled` booleano	Indica se abilitare o meno l'integrazione con il dashboard Operazioni di Cloud Pak for Integration , tramite la traccia. Il valore predefinito è false.
`namespace` Stringa	Spazio dei nomi in cui è installato il dashboard Operazioni di Cloud Pak for Integration .

.spec.tracing.agent

Solo in Cloud Pak for Integration , è possibile configurare le impostazioni per l'agent di traccia facoltativo.

Viene visualizzato in:

- ".spec.tracing" a pagina 140

Campo	Descrizione
`image` Stringa	L'immagine contenitore che verrà utilizzata.
`imagePullPolicy` Stringa	Impostazione che controlla quando il kubelet tenta di estrarre l'immagine specificata. Il valore predefinito è `IfNotPresent`.
`livenessProbe` TracingProbe	Impostazioni che controllano il probe di attività.
`readinessProbe` TracingProbe	Impostazioni che controllano il probe di disponibilità.

.spec.tracing.agent.livenessProbe

Impostazioni che controllano il probe di attività.

Viene visualizzato in:

- ".spec.tracing.agent" a pagina 140

Campo	Descrizione
`failureThreshold` intero	Numero minimo di errori consecutivi perché l'analisi venga considerata non riuscita dopo l'esito positivo. L'impostazione predefinita è 1.

Campo	Descrizione
`initialDelaySeconds` intero	Numero di secondi dopo che il contenitore è stato avviato prima dell'avvio delle analisi di attività. Il valore predefinito è 10 secondi. Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
`periodSeconds` intero	La frequenza (in secondi) con cui eseguire l'analisi. Il valore predefinito è 10 secondi.
`successThreshold` intero	Numero minimo di esiti positivi consecutivi perché il probe venga considerato riuscito dopo l'errore. L'impostazione predefinita è 1.
`timeoutSeconds` intero	Numero di secondi dopo i quali l'analisi va in timeout. Il valore predefinito è 2 secondi. Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.tracing.agent.readinessProbe

Impostazioni che controllano il probe di disponibilità.

Viene visualizzato in:

- ".spec.tracing.agent" a pagina 140

Campo	Descrizione
`failureThreshold` intero	Numero minimo di errori consecutivi perché l'analisi venga considerata non riuscita dopo l'esito positivo. L'impostazione predefinita è 1.
`initialDelaySeconds` intero	Numero di secondi dopo che il contenitore è stato avviato prima dell'avvio delle analisi di attività. Il valore predefinito è 10 secondi. Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
`periodSeconds` intero	La frequenza (in secondi) con cui eseguire l'analisi. Il valore predefinito è 10 secondi.
`successThreshold` intero	Numero minimo di esiti positivi consecutivi perché il probe venga considerato riuscito dopo l'errore. L'impostazione predefinita è 1.
`timeoutSeconds` intero	Numero di secondi dopo i quali l'analisi va in timeout. Il valore predefinito è 2 secondi. Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.tracing.collector

Solo in Cloud Pak for Integration , è possibile configurare le impostazioni per il Tracing Collector facoltativo.

Viene visualizzato in:

- ".spec.tracing" a pagina 140

Campo	Descrizione
`image` Stringa	L'immagine contenitore che verrà utilizzata.
`imagePullPolicy` Stringa	Impostazione che controlla quando il kubelet tenta di estrarre l'immagine specificata. Il valore predefinito è `IfNotPresent`.
`livenessProbe` TracingProbe	Impostazioni che controllano il probe di attività.

Campo	Descrizione
`readinessProbe` TracingProbe	Impostazioni che controllano il probe di disponibilità.

.spec.tracing.collector.livenessProbe

Impostazioni che controllano il probe di attività.

Viene visualizzato in:

- ".spec.tracing.collector" a pagina 141

Campo	Descrizione
`failureThreshold` intero	Numero minimo di errori consecutivi perché l'analisi venga considerata non riuscita dopo l'esito positivo. L'impostazione predefinita è 1.
`initialDelaySeconds` intero	Numero di secondi dopo che il contenitore è stato avviato prima dell'avvio delle analisi di attività. Il valore predefinito è 10 secondi. Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
`periodSeconds` intero	La frequenza (in secondi) con cui eseguire l'analisi. Il valore predefinito è 10 secondi.
`successThreshold` intero	Numero minimo di esiti positivi consecutivi perché il probe venga considerato riuscito dopo l'errore. L'impostazione predefinita è 1.
`timeoutSeconds` intero	Numero di secondi dopo i quali l'analisi va in timeout. Il valore predefinito è 2 secondi. Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.tracing.collector.readinessProbe

Impostazioni che controllano il probe di disponibilità.

Viene visualizzato in:

- ".spec.tracing.collector" a pagina 141

Campo	Descrizione
`failureThreshold` intero	Numero minimo di errori consecutivi perché l'analisi venga considerata non riuscita dopo l'esito positivo. L'impostazione predefinita è 1.
`initialDelaySeconds` intero	Numero di secondi dopo che il contenitore è stato avviato prima dell'avvio delle analisi di attività. Il valore predefinito è 10 secondi. Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.
`periodSeconds` intero	La frequenza (in secondi) con cui eseguire l'analisi. Il valore predefinito è 10 secondi.
`successThreshold` intero	Numero minimo di esiti positivi consecutivi perché il probe venga considerato riuscito dopo l'errore. L'impostazione predefinita è 1.
`timeoutSeconds` intero	Numero di secondi dopo i quali l'analisi va in timeout. Il valore predefinito è 2 secondi. Ulteriori informazioni: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes.

.spec.web

Impostazioni per il server web MQ .

Viene visualizzato in:

- ".spec" a pagina 127

Campo	Descrizione
`enabled` booleano	Indica se abilitare o meno il server Web. Il valore predefinito è false.

.stato

Lo stato osservato di QueueManager.

Viene visualizzato in:

- "QueueManager" a pagina 127

Campo	Descrizione
`adminUiUrl` Stringa	URL per l'interfaccia utente Admin.
`availability` Disponibilità	Lo stato di disponibilità per il gestore code.
Array `conditions` QueueManagerStatusCondition	Le condizioni rappresentano le ultime osservazioni disponibili dello stato del gestore code.
Array `endpoints` QueueManagerStatusEndpoint	Informazioni sugli endpoint che questo gestore code sta esponendo, come gli endpoint API o UI.
`name` Stringa	Il nome del gestore code.
`phase` Stringa	Fase dello stato del gestore code.
`versions` QueueManagerStatusVersion	Versione di MQ utilizzata e altre versioni disponibili da IBM Entitled Registry.

.status.availability

Lo stato di disponibilità per il gestore code.

Viene visualizzato in:

- ".stato" a pagina 143

Campo	Descrizione
`initialQuorumEstablished` booleano	Se è stato stabilito o meno un quorum iniziale per NativeHA.

.status.conditions

QueueManagerStatusCondition definisce le condizioni del Gestore code.

Viene visualizzato in:

- ".stato" a pagina 143

Campo	Descrizione
`lastTransitionTime` Stringa	L'ultima volta in cui la condizione è passata da un stato all'altro.

Campo	Descrizione
`message` Stringa	Messaggio leggibile che indica i dettagli sull'ultima transizione.
`reason` Stringa	Motivo dell'ultima transizione di questo stato.
`status` Stringa	Stato della condizione.
`type` Stringa	Tipo di condizione.

.status.endpoints

QueueManagerStatusEndpoint definisce gli endpoint per QueueManager.

Viene visualizzato in:

- ".stato" a pagina 143

Campo	Descrizione
`name` Stringa	Nome dell'endpoint.
`type` Stringa	Il tipo di endpoint, ad esempio 'UI' per un endpoint UI, 'API' per un endpoint API, 'OpenAPI' per la documentazione API.
`uri` Stringa	URI per l'endpoint.

.status.versions

Versione di MQ utilizzata e altre versioni disponibili da IBM Entitled Registry.

Viene visualizzato in:

- ".stato" a pagina 143

Campo	Descrizione
`available` QueueManagerStatusVersion Disponibile	Altre versioni di MQ disponibili da IBM Entitled Registry.
`reconciled` Stringa	La specifica versione di IBM MQ utilizzata. Se viene specificata un'immagine personalizzata, potrebbe non corrispondere alla versione di MQ effettivamente utilizzata.

.status.versions.available

Altre versioni di MQ disponibili da IBM Entitled Registry.

Viene visualizzato in:

- ".status.versions" a pagina 144

Campo	Descrizione
`channels` schiera	I canali disponibili per l'aggiornamento automatico della versione di MQ .
Array `versions` Versions	Versioni specifiche di MQ disponibili.

.status.versions.available.versions

QueueManagerStatusVersion definisce una versione di MQ.

Viene visualizzato in:

- “.status.versions.available” a pagina 144

Campo	Descrizione
name Stringa	Versione name per questa versione di QueueManager. Questi sono valori validi per il campo spec.version .

OpenShift CP4I *Condizioni di stato per QueueManager (mq.ibm.com/v1beta1)*

I campi **status.conditions** vengono aggiornati per riflettere la condizione della risorsa QueueManager . In generale, le condizioni descrivono situazioni anomale. Un gestore code in uno stato di integrità e pronto non ha condizioni **Error** o **Pending** . Potrebbe avere alcune condizioni **Warning** di avviso.

Il supporto per le condizioni è stato introdotto in IBM MQ Operator 1.2.

Le seguenti condizioni sono state definite per una risorsa QueueManager :

Tabella 1. Condizioni di stato del gestore code

Componente	Tipo di condizione	Codice di errore	Messaggio di avvertenza
QueueManager[7]	In sospeso	Creazione	Il gestore code MQ è in fase di distribuzione
	In sospeso	OidcPending	Il gestore code MQ è in attesa della registrazione del client OIDC
	Errore	Non superato	Distribuzione del gestore code MQ non riuscita
	Avviso	UnsupportedVersion	[8]Un operando è stato installato da un operatore che non è supportato nella versione OCP < *ocp_version>*. Questo operando non è supportato.
	Avviso	Supporto EUS	[9]Un operando EUS < *mq_version>* è stato installato ma è gestito da un operatore che non è qualificato per la durata del supporto esteso. Questo operando non è idoneo per la durata di supporto esteso.
	Avviso	Supporto EUS	[10]È stato installato un operando EUS < *mq_version>* ma OCP versione 4< *ocp_version>* non si qualifica per la durata del supporto esteso. Questo operando non è idoneo per la durata di supporto esteso.
	Avviso	Supporto EUS	[11]È stato installato un operando EUS < *mq_version>* , ma la versione OCP < *ocp_version>* non si qualifica per la durata del supporto esteso. Questo operando è supportato come da una release CD regolare.

[7] Le condizioni `Creating` e `Failed` monitorano l'avanzamento generale della distribuzione del gestore code. Se stai utilizzando una licenza IBM Cloud Pak for Integration e la console web MQ è abilitata, la condizione `OidcPending` registra lo stato del gestore code durante l'attesa del completamento della registrazione del client OIDC con IAM.

Tabella 1. Condizioni di stato del gestore code (Continua)			
Componente	**Tipo di condizione**	**Codice di errore**	**Messaggio di avvertenza**
Pod[12]	In sospeso	PodPending	Il pod per il gestore code MQ è in fase di distribuzione
	Errore	PodFailed	Il pod per il gestore code MQ è in fase di distribuzione
Memoria[13]	In sospeso	StoragePending	È in corso il provisioning della memoria per il gestore code MQ
	Avviso	StorageEphemeral	Utilizzo della memoria temporanea per un gestore code MQ di produzione
	Errore	StorageFailed	Impossibile eseguire il provisioning della memoria per il gestore code MQ

Multi Creazione del tuo contenitore e del tuo codice di distribuzione IBM MQ

Sviluppa un contenitore auto - costruito. Questa è la soluzione del contenitore più flessibile, ma ti richiede di avere forti capacità nella configurazione dei contenitori e di "possedere" il contenitore risultante.

Prima di iniziare

Prima di sviluppare il tuo proprio contenitore, considera se puoi invece utilizzare uno dei contenitori preconfezionati forniti da IBM. Consultare IBM MQ nei contenitori

Informazioni su questa attività

Quando impacchettate IBM MQ come immagine del contenitore, le modifiche alla tua applicazione possono essere distribuite per testare e preparare i sistemi in modo rapido e semplice. Questo può essere un vantaggio importante per la fornitura continua nella tua azienda.

Procedura

- “Pianificazione della tua immagine del gestore code IBM MQ utilizzando un contenitore” a pagina 148

8 Operatore 1.4.0 e successive
9 Operatore 1.4.0 e successive
10 Operatore 1.4.0 e successive
11 Solo operatore 1.3.0
12 Le condizioni del pod monitorano lo stato dei pod durante la distribuzione di un gestore code. Se viene visualizzata una condizione `PodFailed` , anche la condizione generale del gestore code verrà impostata su `Failed`.
13 Le condizioni di archiviazione monitorano l'avanzamento (condizione`StoragePending`) delle richieste per creare volumi per l'archiviazione persistente e riportano errori di bind di ritorno e altri errori. Se si verifica un errore durante il provisioning di memoria, la condizione `StorageFailed` verrà aggiunta all'elenco delle condizioni e anche la condizione generale del gestore code verrà impostata su `Failed`.

- “Creazione di un'immagine del contenitore del gestore code IBM MQ di esempio” a pagina 148
- “Esecuzione di applicazioni di bind locali in contenitori separati” a pagina 151

Concetti correlati
IBM MQ nei contenitori

Multi Pianificazione della tua immagine del gestore code IBM MQ utilizzando un contenitore

Esistono diversi requisiti da considerare quando si esegue un gestore code IBM MQ in un contenitore. L'immagine del contenitore di esempio fornisce un modo per gestire questi requisiti, ma se vuoi utilizzare la tua immagine, devi considerare come vengono gestiti questi requisiti.

Supervisione dei processi

Quando si esegue un contenitore, si sta essenzialmente eseguendo un singolo processo (PID 1 all'interno del contenitore), che può successivamente generare processi child.

Se il processo principale termina, il runtime del contenitore arresta il contenitore. Un gestore code IBM MQ richiede più processi in esecuzione in background.

Per questo motivo, è necessario assicurarsi che il processo principale rimanga attivo finché il gestore code è in esecuzione. Si consiglia di verificare che il gestore code sia attivo da questo processo, ad esempio, eseguendo query amministrative.

Popolamento di `/var/mqm`

I contenitori devono essere configurati con `/var/mqm` come volume.

Quando si esegue questa operazione, la directory del volume è vuota quando il contenitore viene avviato per la prima volta. Questa directory viene generalmente popolata al momento dell'installazione, ma l'installazione e il runtime sono ambienti separati quando si utilizza un contenitore.

Per risolvere questo problema, quando il contenitore viene avviato, puoi utilizzare il comando **`crtmqdir`** per popolare `/var/mqm` quando viene eseguito per la prima volta.

sicurezza contenitore

Per ridurre al minimo i requisiti di sicurezza di runtime, le immagini del contenitore di esempio vengono installate utilizzando l'installazione dezippabile IBM MQ . Ciò garantisce che non sia impostato alcun bit `setuid` e che il contenitore non debba utilizzare l'escalation dei privilegi. Alcuni sistemi di contenitori definiscono quali ID utente sono in grado di utilizzare e l'installazione non zippabile non fa alcun presupposto sugli utenti del sistema operativo disponibili.

Multi Creazione di un'immagine del contenitore del gestore code IBM MQ di esempio

Utilizzare queste informazioni per creare un'immagine del contenitore di esempio per l'esecuzione di un gestore code IBM MQ in un contenitore.

Informazioni su questa attività

Innanzitutto, si crea un'immagine di base contenente un file system Red Hat Universal Base Image e un'installazione pulita di IBM MQ.

In secondo luogo, crei un altro livello di immagine del contenitore sopra la base, che aggiunge alcune configurazioni IBM MQ per consentire la sicurezza di ID utente e password di base.

Infine, si esegue un contenitore che utilizza questa immagine come file system, con il contenuto di `/var/mqm` fornito da un volume specifico del contenitore sul file system host.

Procedura

- Per informazioni su come creare un'immagine del contenitore di esempio per l'esecuzione di un gestore code IBM MQ in un contenitore, consultare i seguenti argomenti secondari:
  - “Creazione di un'immagine del gestore code IBM MQ di base di esempio” a pagina 149
  - “Creazione di un'immagine del gestore code IBM MQ configurata di esempio” a pagina 149

Multi Creazione di un'immagine del gestore code IBM MQ di base di esempio

Per utilizzare IBM MQ nella tua immagine contenitore, devi inizialmente creare un'immagine base con un'installazione IBM MQ pulita. La seguente procedura ti mostra come creare un'immagine base di esempio, utilizzando il codice di esempio ospitato su GitHub.

Procedura

- Utilizza i file make forniti nel repository mq - container GitHub per creare la tua immagine contenitore di produzione.

  Attieniti alle istruzioni in Creazione di un'immagine del contenitore su GitHub. Se si prevede di configurare l'accesso sicuro utilizzando l'SCC (Security Context Constraint) Red Hat OpenShift Container Platform "limitato" , è necessario utilizzare il pacchetto 'No-Install' IBM MQ .

Risultati

Hai ora un'immagine del contenitore di base con IBM MQ installato.

Ora è possibile creare un'immagine del gestore code IBM MQ configurata di esempio.

Multi Creazione di un'immagine del gestore code IBM MQ configurata di esempio

Dopo aver creato la tua immagine del contenitore IBM MQ di base generica, devi applicare la tua propria configurazione per consentire l'accesso sicuro. A tale scopo, si crea un proprio livello di immagine contenitore, utilizzando l'immagine generica come elemento principale.

Prima di iniziare

V 9.2.0 Questa attività presuppone che, quando si crea l'immagine del gestore code IBM MQ di esempio, sia stato utilizzato il package "Nessuna installazione" IBM MQ . In caso contrario, non è possibile configurare l'accesso protetto utilizzando l'SCC (Security Context Constraint) Red Hat OpenShift Container Platform "limitato" . L'SCC "limitato" , che viene utilizzato per impostazione predefinita, utilizza ID utente casuali e impedisce l'escalation dei privilegi passando a un utente differente. Il programma di installazione IBM MQ tradizionale basato su RPM si basa su un utente e un gruppo `mqm` e utilizza anche bit `setuid` su programmi eseguibili. In IBM MQ 9.2, quando si utilizza il pacchetto "No - Install" IBM MQ , non esiste più alcun utente `mqm` né un gruppo `mqm` .

Procedura

1. Creare una nuova directory e aggiungere un file denominato `config.mqsc`, con il seguente contenuto:

```
DEFINE QLOCAL(EXAMPLE.QUEUE.1) REPLACE
```

   Tenere presente che l'esempio precedente utilizza l'autenticazione semplice di ID utente e password. Tuttavia, è possibile applicare qualsiasi configurazione di sicurezza richiesta dalla propria azienda.

2. Creare un file denominato `Dockerfile`, con il seguente contenuto:

```
FROM mq
COPY config.mqsc /etc/mqm/
```

3. Crea la tua immagine contenitore personalizzata utilizzando il seguente comando:

```
docker build -t mymq .
```

dove "." è la directory contenente i due file appena creati.

Docker crea quindi un container temporaneo utilizzando tale immagine ed esegue i restanti comandi.

Nota: Su Red Hat Enterprise Linux (RHEL), utilizzi il comando **docker** (RHEL V7) o **podman** (RHEL V7 o RHEL V8). Su Linux, sarà necessario eseguire i comandi **docker** con **sudo** all'inizio del comando, per ottenere ulteriori privilegi.

4. Esegui la tua nuova immagine personalizzata per creare un nuovo contenitore, con l'immagine disco che hai appena creato.

Il nuovo livello immagine non ha specificato alcun comando particolare da eseguire, quindi è stato ereditato dall'immagine principale. Il punto di immissione del parent (il codice è disponibile su GitHub):

- Crea un gestore code
- Avvia il gestore code
- Crea un listener predefinito
- Quindi, esegue i comandi MQSC da `/etc/mqm/config.mqsc`.

Immetti i comandi seguenti per eseguire la nuova immagine personalizzata:

```
docker run \
  --env LICENSE=accept \
  --env MQ_QMGR_NAME=QM1 \
  --volume /var/example:/var/mqm \
  --publish 1414:1414 \
  --detach \
  mymq
```

dove:

Primo parametro env
Passa una variabile di ambiente nel contenitore, che riconosce l'accettazione della licenza per IBM IBM WebSphere MQ. È anche possibile impostare la variabile LICENSE da visualizzare per visualizzare la licenza.

Consultare IBM MQ informazioni sulla licenza per ulteriori dettagli sulle licenze IBM MQ .

Secondo parametro env
Imposta il nome del gestore code che si utilizza.

Parametro volume
Indica al contenitore che qualsiasi cosa MQ scriva in `/var/mqm` deve essere effettivamente scritta in `/var/example` sull'host.

Questa opzione significa che è possibile eliminare facilmente il contenitore in un secondo momento e conservare ancora i dati persistenti. Questa opzione rende inoltre più semplice la visualizzazione dei file di log.

Pubblica parametro
Associa le porte sul sistema host alle porte nel contenitore. Il contenitore viene eseguito per impostazione predefinita con il suo indirizzo IP interno, il che significa che devi associare in modo specifico tutte le porte che vuoi esporre.

In questo esempio, ciò significa associare la porta 1414 sull'host alla porta 1414 nel contenitore.

Scollega parametro
Esegue il contenitore in background.

Risultati

Hai creato un'immagine del contenitore configurata e puoi visualizzare i contenitori in esecuzione utilizzando il comando **docker ps** . Puoi visualizzare i processi IBM MQ in esecuzione nel tuo contenitore utilizzando il comando **docker top** .

Attenzione:

Puoi visualizzare i log di un contenitore utilizzando il comando **docker logs ${CONTAINER_ID}** .

Operazioni successive

- Se il tuo contenitore non viene visualizzato quando utilizzi il comando **docker ps** , il contenitore potrebbe non essere riuscito. Puoi visualizzare i contenitori non riusciti utilizzando il comando **docker ps -a** .
- Quando utilizzi il comando **docker ps -a** , viene visualizzato l'ID contenitore. Questo ID è stato stampato anche quando è stato immesso il comando **docker run** .
- È possibile visualizzare i log di un contenitore utilizzando il comando **docker logs ${CONTAINER_ID}** .

Multi Esecuzione di applicazioni di bind locali in contenitori separati

Con la condivisione dello spazio dei nomi del processo tra contenitori in Docker, è possibile eseguire le applicazioni che richiedono una connessione di bind locale a IBM MQ in contenitori separati dal gestore code IBM MQ .

Informazioni su questa attività

Questa funzione è supportata in IBM MQ 9.0.3 e nei gestori code successivi.

È necessario rispettare le seguenti condizioni:

- Devi condividere lo spazio dei nomi PID dei contenitori utilizzando l'argomento `--pid` .
- Devi condividere lo spazio dei nomi IPC dei contenitori utilizzando l'argomento `--ipc` .
- È necessario:
  1. Condividere lo spazio dei nomi UTS dei contenitori con l'host utilizzando l'argomento `--uts` oppure
  2. Verificare che i contenitori abbiano lo stesso nome host utilizzando l'argomento `-h` o `--hostname` .
- È necessario montare la directory di dati IBM MQ in un volume disponibile per tutti i contenitori nella directory `/var/mqm` .

È possibile provare questa funzionalità completando la seguente procedura su un sistema Linux su cui è già installato Docker .

Il seguente esempio utilizza l'immagine del contenitore IBM MQ di esempio. Puoi trovare i dettagli di questa immagine su Github.

Procedura

1. Creare una directory temporanea che agisca come volume, immettendo il seguente comando:

```
mkdir /tmp/dockerVolume
```

2. Creare un gestore code (QM1) in un contenitore, denominato `sharedNamespace`, immettendo il seguente comando:

```
docker run -d -e LICENSE=accept -e MQ_QMGR_NAME=QM1 --volume /tmp/dockerVol:/mnt/mqm
--uts host --name sharedNamespace ibmcom/mq
```

3. Avviare un secondo contenitore denominato `secondaryContainer`, basato sul `ibmcom/mq`, ma non creare un gestore code immettendo il seguente comando:

```
docker run --entrypoint /bin/bash --volumes-from sharedNamespace --pid
container:sharedNamespace --ipc container:sharedNamespace --uts host --name
secondaryContainer -it --detach ibmcom/mq
```

4. Eseguire il comando **dspmq** sul secondo contenitore, per visualizzare lo stato di entrambi i gestori code, immettendo il seguente comando:

```
docker exec secondaryContainer dspmq
```

5. Eseguire il seguente comando per elaborare i comandi MQSC rispetto al gestore code in esecuzione sull'altro contenitore:

```
docker exec -it secondaryContainer runmqsc QM1
```

Risultati

Ora le applicazioni locali sono in esecuzione in contenitori separati e ora è possibile eseguire correttamente comandi come **dspmq**, **amqsput**, **amqsget**e **runmqsc** come bind locali al gestore code QM1 dal contenitore secondario.

Se non viene visualizzato il risultato previsto, consultare “Risoluzione dei problemi delle applicazioni dello spazio dei nomi” a pagina 152 per ulteriori informazioni.

Multi Risoluzione dei problemi delle applicazioni dello spazio dei nomi

Quando si utilizzano spazi dei nomi condivisi, è necessario assicurarsi di condividere tutti gli spazi dei nomi (IPC, PID e UTS/nomehost) e i volumi montati, altrimenti le applicazioni non funzioneranno.

Consultare “Esecuzione di applicazioni di bind locali in contenitori separati” a pagina 151 per un elenco delle limitazioni da seguire.

Se la tua applicazione non soddisfa tutte le limitazioni elencate, potresti riscontrare problemi nel punto in cui viene avviato il contenitore, ma la funzionalità prevista non funziona.

Il seguente elenco delinea alcune cause comuni e il comportamento che si sta probabilmente osservando se si è dimenticato di rispettare una delle restrizioni.

- Se dimentichi di condividere lo spazio dei nomi (UTS / PID/IPC) o il nome host dei contenitori e monti il volume, il tuo contenitore sarà in grado di vedere il gestore code ma non di interagire con il gestore code.
  - Per i comandi **dspmq** , vedi quanto segue:

```
docker exec container dspmq

QMNAME(QM1)                       STATUS(Status not available)
```

  - Per i comandi **runmqsc** o altri comandi che tentano di connettersi al gestore code, è probabile che si riceva un messaggio di errore 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
```

- Se condividi tutti gli spazi dei nomi richiesti ma non monti un volume condiviso nella directory `/var/mqm` e hai un percorso dati IBM MQ valido, i tuoi comandi ricevono anche i messaggi di errore AMQ8146 .

Tuttavia, **dspmq** non è in grado di visualizzare il tuo gestore code e restituisce invece una risposta vuota:

```
docker exec container dspmq
```

- Se si condividono tutti gli spazi dei nomi richiesti ma non si monta un volume condiviso nella directory `/var/mqm` e non si dispone di un percorso dati IBM MQ valido (o nessun percorso dati IBM MQ), vengono visualizzati diversi errori poiché il percorso dati è un componente chiave di un'installazione IBM MQ . Senza il percorso dati, IBM MQ non può funzionare.

  Se si esegue uno dei seguenti comandi e vengono visualizzate risposte simili a quelle mostrate in questi esempi, è necessario verificare di aver montato la directory o di aver creato una directory di dati 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 Creazione del gruppo HA nativo se si creano i propri contenitori

È necessario creare, configurare e avviare tre gestori code per creare il gruppo HA nativo.

Informazioni su questa attività

Il metodo consigliato per la creazione di una soluzione HA nativa è l'utilizzo dell'operatore IBM MQ (consultare HA nativa). In alternativa, se crei i tuoi contenitori, puoi seguire queste istruzioni.

Per creare un gruppo HA nativo, creare tre gestori code su tre nodi con il tipo di log impostato su `log replication`. Si modifica quindi il file `qm.ini` per ciascun gestore code per aggiungere i dettagli di connessione per ognuno dei tre nodi in modo che possano replicare i dati di log l'uno sull'altro.

È necessario, quindi, avviare tutti e tre i gestori code in modo che possano controllare che tutte e tre le istanze possano comunicare tra loro e determinare quale di essi sarà l'istanza attiva e quali saranno le replica.

Procedura

1. Su ognuno dei tre nodi, creare un gestore code, specificando un tipo di log di replica del log e fornendo un nome univoco per ciascuna istanza del log. Ogni gestore code ha lo stesso nome:

   ```
   crtmqm -lr instance_name qmname
   ```

   Ad esempio:

```
node 1> crtmqm -lr qm1_inst1 qm1

node 2> crtmqm -lr qm1_inst2 qm1

node 3> crtmqm -lr qm1_inst3 qm1
```

2. Una volta creato correttamente ciascun gestore code, una stanza aggiuntiva denominata `NativeHALocalInstance` viene aggiunta al file di configurazione del gestore code, `qm.ini`. Un attributo `Name` viene aggiunto alla sezione specificando il nome istanza fornito.

   È possibile aggiungere facoltativamente i seguenti attributi alla sezione `NativeHALocalInstance` nel file `qm.ini` :

   KeyRepository
   L'ubicazione del repository delle chiavi che contiene il certificato digitale da utilizzare per la protezione del traffico di replica del log. L'ubicazione viene fornita in formato di radice, ossia include il percorso completo e il nome file senza un'estensione. Se l'attributo della sezione `KeyRepository` viene omesso, i dati di replica del log vengono scambiati tra le istanze in testo semplice.

   CertificateLabel
   L'etichetta del certificato che indica il certificato digitale da utilizzare per la protezione del traffico di replica del log. Se `KeyRepository` viene fornito ma `CertificateLabel` viene omesso, viene utilizzato il valore predefinito `ibmwebspheremq`*queue_manager* .

   CipherSpec
   La CipherSpec di MQ da utilizzare per proteggere il traffico di replica del log. Se questo attributo della stanza viene fornito, è necessario fornire anche `KeyRepository` . Se `KeyRepository` viene fornito ma `CipherSpec` viene omesso, viene utilizzato il valore predefinito `ANY` .

   LocalAddress
   L'indirizzo dell'interfaccia di rete locale che accetta il traffico di replica del log. Se questo attributo della stanza viene fornito, identifica l'interfaccia di rete locale e / o la porta utilizzando il formato "[addr] [(port)]". L'indirizzo di rete può essere specificato come un nome host, in formato decimale puntato IPv4 o in formato esadecimale IPv6 . Se questo attributo viene omesso, il gestore code tenta di collegarsi a tutte le interfacce di rete, utilizza la porta specificata in `ReplicationAddress` nella stanza `NativeHAInstances` che corrisponde al nome dell'istanza locale.

   HeartbeatInterval
   L'intervallo di heartbeat definisce la frequenza in millisecondi con cui un'istanza attiva di un gestore code HA nativo invia un heartbeat di rete. L'intervallo valido del valore dell'intervallo di heartbeat è compreso tra 500 (0.5 secondi) e 60000 (1 minuto), un valore esterno a questo intervallo causa l'errore di avvio del gestore code. Se questo attributo viene omesso, viene utilizzato un valore predefinito di 5000 (5 secondi). Ogni istanza deve utilizzare lo stesso intervallo di heartbeat.

   HeartbeatTimeout
   Il timeout heartbeat definisce il tempo di attesa di un'istanza di replica di un gestore code HA nativo prima di decidere che l'istanza attiva non risponde. L'intervallo valido del valore di timeout dell'intervallo di heartbeat è compreso tra 500 (0.5 secondi) e 120000 (2 minuti). Il valore del timeout di heartbeat deve essere maggiore o uguale all'intervallo di heartbeat.

   Un valore non valido causa l'errore di avvio del gestore code. Se questo attributo viene omesso, una replica attende 2 x `HeartbeatInterval` prima di avviare il processo per selezionare una nuova istanza attiva. Ogni istanza deve utilizzare lo stesso timeout heartbeat.

   RetryInterval
   L'intervallo di nuovi tentativi definisce la frequenza in millisecondi con cui un gestore code HA nativo deve ritentare un link di replica non riuscito. L'intervallo valido per i tentativi è compreso tra 500 (0.5 secondi) e 120000 (2 minuti). Se questo attributo viene omesso, una replica attende 2 x `HeartbeatInterval` prima di ritentare un link di replica non riuscito.

3. Modificare il file `qm.ini` per ogni gestore code e aggiungere dettagli di connessione. Si aggiungono tre stanze `NativeHAInstance` , una per ogni istanza del gestore code nel gruppo HA nativo (inclusa l'istanza locale). Aggiungere i seguenti attributi:

   Nome
   Specificare il nome istanza utilizzato quando è stata creata l'istanza del gestore code.

   ReplicationAddress
   Specificare il nome host, IPv4 decimale puntato o IPv6 l'indirizzo in formato esadecimale dell'istanza. È possibile specificare l'indirizzo come nome host, IPv4 decimale puntato o IPv6 indirizzo in formato esadecimale. L'indirizzo di replica deve essere risolvibile e instradabile da ogni istanza nel gruppo. Il numero di porta da utilizzare per la replica del log deve essere specificato tra parentesi, ad esempio:

   ```
   ReplicationAddress=host1.example.com(4444)
   ```

   Nota: Le stanze `NativeHAInstance` sono identiche in ogni istanza e possono essere fornite utilizzando la configurazione automatica (**`crtmqm -ii`**).
4. Avviare ciascuna delle seguenti tre istanze:

   ```
   strmqm QMgrName
   ```

   Quando le istanze vengono avviate, comunicano per controllare che tutte e tre le istanze siano in esecuzione, quindi decidere quale delle tre è l'istanza attiva, mentre le altre due istanze continuano ad eseguire come repliche.

Esempio

Il seguente esempio mostra la sezione di un file `qm.ini` che specifica i dettagli della HA nativa richiesti per una delle tre istanze:

```
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 Considerazioni sull'esecuzione di un aggiornamento continuo di un gestore code HA nativo

Qualsiasi aggiornamento alla versione o alla specifica Pod di IBM MQ per un gestore code HA nativo richiederà l'esecuzione di un aggiornamento progressivo delle istanze del gestore code. Il IBM MQ Operator lo gestisce automaticamente, ma se stai creando il tuo codice di distribuzione, ci sono alcune considerazioni importanti.

Nota: Il grafico Helm di esempio include uno script di shell per eseguire un aggiornamento progressivo, ma lo script **non** è adatto per l'utilizzo in produzione, poiché non affronta le considerazioni in questo argomento.

In Kubernetes `StatefulSet` , le risorse vengono utilizzate per gestire l'avvio ordinato e gli aggiornamenti continui. Parte della procedura di avvio è quella di avviare ogni Pod individualmente, attendere che diventi pronto, e quindi passare al Pod successivo. Questo non funzionerà per l'HA nativa, poiché tutti i Pods devono essere avviati in modo che possano eseguire un'elezione di leader. Pertanto, il campo `.spec.podManagementPolicy` su `StatefulSet` deve essere impostato su `Parallel`. Ciò significa anche che tutti i pod saranno aggiornati in parallelo, il che è particolarmente indesiderabile. Per questo motivo, `StatefulSet` deve utilizzare anche la strategia di aggiornamento `OnDelete` .

L'impossibilità di utilizzare il codice di aggiornamento continuo `StatefulSet` rende necessario il codice di aggiornamento continuo personalizzato, che dovrebbe considerare quanto segue:

- Procedura generale di aggiornamento a rotazione
- Ridurre al minimo i tempi di inattività aggiornando i pod nell'ordine migliore
- Gestione delle modifiche nello stato del cluster
- Gestione degli errori
- Gestione dei problemi di temporizzazione

Procedura generale di aggiornamento a rotazione

Il codice di aggiornamento progressivo deve attendere che ogni istanza mostri uno stato di `REPLICA` da `dspmq`. Ciò significa che l'istanza ha eseguito un certo livello di avvio (ad esempio, il contenitore è avviato e i processi MQ sono in esecuzione), ma non è necessariamente riuscita a comunicare con le altre istanze. Ad esempio: il Pod A viene riavviato e non appena è in stato `REPLICA` , il Pod B viene riavviato. Una volta che il Pod B inizia con la nuova configurazione, dovrebbe essere in grado di parlare con il Pod A, e può formare il quorum, e A o B diventeranno la nuova istanza attiva.

Come parte di questo, è utile avere un ritardo dopo che ogni pod ha raggiunto lo stato `REPLICA` , per consentirgli di connettersi ai suoi peer e stabilire il quorum.

Ridurre al minimo i tempi di inattività aggiornando i pod nell'ordine migliore

Il codice di aggiornamento continuo deve eliminare i pod uno alla volta, iniziando con i pod che si trovano in un stato di errore noto, seguiti da tutti i pod che non sono stati avviati correttamente. Il pod del gestore code attivo deve essere generalmente aggiornato per ultimo.

È anche importante mettere in pausa l'eliminazione dei pod se l'ultimo aggiornamento ha portato un pod in uno stato di errore noto. Ciò impedisce il roll-out di un aggiornamento interrotto su tutti i pod. Ad esempio, questo può accadere se il pod viene aggiornato per utilizzare una nuova immagine del contenitore che non è accessibile (o contiene un errore di battitura).

Gestione delle modifiche nello stato del cluster

Il codice di aggiornamento progressivo deve reagire in maniera appropriata alle modifiche in tempo reale nello stato del cluster. Ad esempio, uno dei pod del gestore code potrebbe essere eliminato a seguito di un riavvio del nodo o a causa della pressione del nodo. È possibile che un pod sfrattato non venga immediatamente ripianificato se il cluster è occupato. In questo caso, il codice di aggiornamento scorrevole dovrebbe attendere in modo appropriato prima di riavviare qualsiasi altro pod.

Gestione degli errori

Il codice di aggiornamento continuo deve essere robusto per gli errori quando si richiama l'API Kubernetes e un altro comportamento del cluster non previsto.

Inoltre, il codice di aggiornamento continuo deve essere tollerante al riavvio. Un aggiornamento a rotazione può essere di lunga durata e potrebbe essere necessario riavviare il codice.

Gestione dei problemi di temporizzazione

Il codice di aggiornamento continuo deve controllare le revisioni di aggiornamento del pod, in modo che possa garantire che il pod sia stato riavviato. Ciò evita problemi di sincronizzazione in cui un pod può indicare che è "Avviato", ma in realtà non è ancora terminato.

Concetti correlati

“Scelta della modalità di utilizzo di IBM MQ nei contenitori” a pagina 5
Ci sono diverse opzioni per l'utilizzo di IBM MQ nei contenitori: puoi scegliere di utilizzare IBM MQ Operator, che utilizza le immagini del contenitore preconfezionate, oppure puoi creare le tue immagini e il tuo codice di distribuzione.

CP4I Visualizzazione dello stato dei gestori code della HA nativa per i contenitori personalizzati

Per i contenitori personalizzati, puoi visualizzare lo stato delle istanze Native HA utilizzando il comando **dspmq** .

Informazioni su questa attività

È possibile utilizzare il comando **dspmq** per visualizzare lo stato operativo di un'istanza del gestore code su un nodo. Le informazioni restituite dipendono dal fatto che l'istanza sia attiva o una replica. Le informazioni fornite dall'istanza attiva sono definitive, le informazioni dai nodi di replica potrebbero non essere aggiornate.

È possibile effettuare le seguenti azioni:

- Visualizzare se l'istanza del gestore code sul nodo corrente è attiva o una replica.
- Visualizza lo stato operativo della HA nativa dell'istanza sul nodo corrente.
- Visualizzare lo stato operativo di tutte e tre le istanze in una configurazione HA nativa.

I seguenti campi di stato vengono utilizzati per riportare lo stato di configurazione della HA nativa:

Ruolo
Specifica il ruolo corrente dell'istanza ed è uno tra `Active`, `Replica`o `Unknown`.

ISTANZA
Il nome fornito per questa istanza del gestore code quando è stata creata utilizzando l'opzione **-lr** del comando **crtmqm** .

INSYNC
Indica se l'istanza è in grado di assumere il controllo come istanza attiva, se richiesto.

Quorum
Riporta lo stato del quorum nel formato *number_of_instances_in - sync/ number_of_instances_configured*.

REPLADDR
L'indirizzo di replica dell'istanza del gestore code.

COLLEGA
Indica se il nodo è connesso all'istanza attiva.

BACKLOG
Indica il numero di KB in cui si trova l'istanza.

CONNETTIN
Indica se l'istanza denominata è connessa a questa istanza.

ALTDATE
Indica la data in cui queste informazioni sono state aggiornate l'ultima volta (vuoto se non sono mai state aggiornate).

ALTTIME
Indica l'ora in cui queste informazioni sono state aggiornate l'ultima volta (vuoto se non sono mai state aggiornate).

Procedura

- Per determinare se un'istanza del gestore code è in esecuzione come istanza attiva o come replica:

```
dspmq -o status -m QMgrName
```

Un'istanza attiva di un gestore code denominato BOB riporta il seguente stato:

```
QMNAME(BOB)                 STATUS(Running)
```

Un'istanza di replica di un gestore code denominato BOB riporta il seguente stato:

```
QMNAME(BOB)                STATUS(Replica)
```

Un'istanza inattiva riporta il seguente stato:

```
QMNAME(BOB)                STATUS(Ended Immediately)
```

- Per determinare lo stato operativo della HA nativa dell'istanza sul nodo corrente:

```
dspmq -o nativeha -m QMgrName
```

L'istanza attiva di un gestore code denominato BOB potrebbe riportare il seguente stato:

```
QMNAME(BOB)               ROLE(Active) INSTANCE(inst1) INSYNC(Yes) QUORUM(3/3)
```

Un'istanza di replica di un gestore code denominato BOB potrebbe riportare il seguente stato:

```
QMNAME(BOB)               ROLE(Replica) INSTANCE(inst2) INSYNC(Yes) QUORUM(2/3)
```

Un'istanza inattiva di un gestore code denominato BOB potrebbe riportare il seguente stato:

```
QMNAME(BOB)               ROLE(Unknown) INSTANCE(inst3) INSYNC(no) QUORUM(0/3)
```

- Per determinare lo stato operativo della HA nativo di tutte le istanze nella configurazione della HA nativa:

```
dspmq -o nativeha -x -m QMgrName
```

Se si immette questo comando sul nodo che esegue l'istanza attiva del BOB del gestore code, è possibile che si riceva il seguente stato:

```
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)
```

Se si immette questo comando su un nodo che esegue un'istanza di replica del BOB del gestore code, è possibile che si riceva il seguente stato, che indica che una delle repliche è in ritardo:

```
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)
```

Se si immette questo comando su un nodo che esegue un'istanza inattiva del BOB del gestore code, è possibile che si riceva il seguente stato:

```
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()
```

Se si immette il comando quando le istanze stanno ancora negoziando quali sono attive e quali sono repliche, si riceverà il seguente stato:

```
QMNAME(BOB)              STATUS(Negotiating)
```

Riferimenti correlati
dspmq

CP4I Chiusura gestori code della HA nativa

È possibile utilizzare il comando **endmqm** per terminare un gestore code attivo o di replica che fa parte di un gruppo HA nativo.

Procedura

- Per terminare l''istanza attiva di un gestore code, fare riferimento a Fine dei gestori code della HA nativa nella sezione Configurazione di questa documentazione.

Informazioni particolari

Queste informazioni sono state sviluppate per i prodotti ed i servizi offerti negli Stati Uniti.

IBM potrebbe non offrire i prodotti, i servizi o le funzioni descritti in questo documento in altri paesi. Consultare il rappresentante IBM locale per informazioni sui prodotti e sui servizi disponibili nel proprio paese. Ogni riferimento relativo a prodotti, programmi o servizi IBM non implica che solo quei prodotti, programmi o servizi IBM possano essere utilizzati. In sostituzione a quelli forniti da IBM possono essere usati prodotti, programmi o servizi funzionalmente equivalenti che non comportino la violazione dei diritti di proprietà intellettuale o di altri diritti dell'IBM. È comunque responsabilità dell'utente valutare e verificare la possibilità di utilizzare altri programmi e/o prodotti, fatta eccezione per quelli espressamente indicati dall'IBM.

IBM potrebbe disporre di applicazioni di brevetti o brevetti in corso relativi all'argomento descritto in questo documento. La fornitura di tale documento non concede alcuna licenza a tali brevetti. Chi desiderasse ricevere informazioni relative a licenze può rivolgersi per iscritto a:

Director of Commercial Relations
IBM Corporation
Schoenaicher Str. 220
D-7030 Boeblingen
U.S.A.

Per richieste di licenze relative ad informazioni double-byte (DBCS), contattare il Dipartimento di Proprietà Intellettuale IBM nel proprio paese o inviare richieste per iscritto a:

Intellectual Property Licensing
Legge sulla proprietà intellettuale e legale
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

Il seguente paragrafo non si applica al Regno Unito o a qualunque altro paese in cui tali dichiarazioni sono incompatibili con le norme locali: INTERNATIONAL BUSINESS MACHINES CORPORATION FORNISCE LA PRESENTE PUBBLICAZIONE "NELLO STATO IN CUI SI TROVA" SENZA GARANZIE DI ALCUN TIPO, ESPRESSE O IMPLICITE, IVI INCLUSE, A TITOLO DI ESEMPIO, GARANZIE IMPLICITE DI NON VIOLAZIONE, DI COMMERCIABILITÀ E DI IDONEITÀ PER UNO SCOPO PARTICOLARE. Alcuni stati non consentono la rinuncia a garanzie esplicite o implicite in determinate transazioni; quindi la presente dichiarazione potrebbe non essere applicabile.

Questa pubblicazione potrebbe contenere imprecisioni tecniche o errori tipografici. Le informazioni incluse in questo documento vengono modificate su base periodica; tali modifiche vengono incorporate nelle nuove edizioni della pubblicazione. IBM si riserva il diritto di apportare miglioramenti o modifiche al prodotto/i e/o al programma/i descritti nella pubblicazione in qualsiasi momento e senza preavviso.

Qualsiasi riferimento a siti Web non IBM contenuto nelle presenti informazioni è fornito per consultazione e non vuole in alcun modo promuovere i suddetti siti Web. I materiali presenti in tali siti Web non sono parte dei materiali per questo prodotto IBM e l'utilizzo di tali siti Web è a proprio rischio.

Tutti i commenti e i suggerimenti inviati potranno essere utilizzati liberamente da IBM e diventeranno esclusiva della stessa.

Coloro che detengono la licenza su questo programma e desiderano avere informazioni su di esso allo scopo di consentire (i) uno scambio di informazioni tra programmi indipendenti ed altri (compreso questo) e (ii) l'uso reciproco di tali informazioni, dovrebbero rivolgersi a:

IBM Corporation
Coordinatore interoperabilità software, Dipartimento 49XA
Autostrada 3605 52 N

Rochester, MN 55901
U.S.A.

Queste informazioni possono essere rese disponibili secondo condizioni contrattuali appropriate, compreso, in alcuni casi, il pagamento di un addebito.

Il programma su licenza descritto in queste informazioni e tutto il materiale su licenza disponibile per esso sono forniti da IBM in termini di IBM Customer Agreement, IBM International Program License Agreement o qualsiasi altro accordo equivalente tra le parti.

Tutti i dati relativi alle prestazioni contenuti in questo documento sono stati determinati in un ambiente controllato. Pertanto, i risultati ottenuti in altri ambienti operativi possono variare in modo significativo. Alcune misurazioni potrebbero essere state fatte su sistemi a livello di sviluppo e non vi è alcuna garanzia che queste misurazioni saranno le stesse sui sistemi generalmente disponibili. Inoltre, alcune misurazioni potrebbero essere state stimate mediante estrapolazione. I risultati quindi possono variare. Gli utenti di questo documento dovrebbero verificare i dati applicabili per il loro ambiente specifico.

Le informazioni relative a prodotti non IBM provengono dai fornitori di tali prodotti, dagli annunci pubblicati o da altre fonti pubblicamente disponibili. IBM non ha verificato tali prodotti e, pertanto, non può garantirne l'accuratezza delle prestazioni. Eventuali commenti relativi alle prestazioni dei prodotti non IBM devono essere indirizzati ai fornitori di tali prodotti.

Tutte le dichiarazioni riguardanti la direzione o l'intento futuro di IBM sono soggette a modifica o ritiro senza preavviso e rappresentano solo scopi e obiettivi.

Questa pubblicazione contiene esempi di dati e prospetti utilizzati quotidianamente nelle operazioni aziendali, Per illustrarle nel modo più completo possibile, gli esempi includono i nomi di individui, società, marchi e prodotti. Tutti questi nomi sono fittizi e qualsiasi somiglianza con nomi ed indirizzi adoperati da imprese realmente esistenti sono una mera coincidenza.

LICENZA SUL COPYRIGHT:

Queste informazioni contengono programmi applicativi di esempio in lingua originale, che illustrano le tecniche di programmazione su diverse piattaforme operative. È possibile copiare, modificare e distribuire questi programmi di esempio sotto qualsiasi forma senza alcun pagamento alla IBM, allo scopo di sviluppare, utilizzare, commercializzare o distribuire i programmi applicativi in conformità alle API (application programming interface) a seconda della piattaforma operativa per cui i programmi di esempio sono stati scritti. Questi esempi non sono stati testati approfonditamente tenendo conto di tutte le condizioni possibili. IBM, quindi, non può garantire o sottintendere l'affidabilità, l'utilità o il funzionamento di questi programmi.

Se si sta visualizzando queste informazioni in formato elettronico, le fotografie e le illustrazioni a colori potrebbero non apparire.

Informazioni sull'interfaccia di programmazione

Le informazioni sull'interfaccia di programmazione, se fornite, consentono di creare software applicativo da utilizzare con questo programma.

Questo manuale contiene informazioni sulle interfacce di programmazione che consentono al cliente di scrivere programmi per ottenere i servizi di WebSphere MQ.

Queste informazioni, tuttavia, possono contenere diagnosi, modifica e regolazione delle informazioni. La diagnosi, la modifica e la regolazione delle informazioni vengono fornite per consentire il debug del software applicativo.

Importante: Non utilizzare queste informazioni di diagnosi, modifica e ottimizzazione come interfaccia di programmazione perché sono soggette a modifica.

Marchi

IBM, il logo IBM , ibm.com, sono marchi di IBM Corporation, registrati in molte giurisdizioni nel mondo. Un elenco aggiornato dei marchi IBM è disponibile sul web in "Copyright and trademark

information"www.ibm.com/legal/copytrade.shtml. Altri nomi di prodotti e servizi potrebbero essere marchi di IBM o altre società.

Microsoft e Windows sono marchi di Microsoft Corporation negli Stati Uniti e/o in altri paesi.

UNIX è un marchio registrato di The Open Group negli Stati Uniti e/o in altri paesi.

Linux è un marchi registrato di Linus Torvalds negli Stati Uniti e/o in altri paesi.

Questo prodotto include il software sviluppato da Eclipse Project (https://www.eclipse.org/).

Java e tutti i marchi e i logo Java sono marchi registrati di Oracle e/o di società affiliate.

Numero parte:

(1P) P/N: