9.4

IBM MQ nei contenitori

Nota

Prima di utilizzare queste informazioni e il prodotto che supportano, leggere le informazioni in “Informazioni particolari” a pagina 299.

Questa edizione si applica alla versione 9 release 4 di IBM® MQ e a tutte le successive release e modifiche se non diversamente indicato nelle nuove edizioni.

Indice

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.

Informazioni su IBM MQ nei contenitori

Informazioni introduttive per aiutarti a iniziare a utilizzare IBM MQ nei contenitori.

I contenitori sono una tecnologia per consentire l'impacchettamento e l'isolamento del codice con il suo ambiente di runtime, che può essere eseguito in modo isolato da altri software sulla stessa infrastruttura. Ciò semplifica lo spostamento di un gestore code o di un'applicazione tra ambienti (ad esempio, sviluppo, test e produzione). I moderni orchestratori di contenitori, come Red Hat OpenShift Container Platform e Kubernetes , possono eseguire molti tipi di contenitori sulla stessa macchina, ognuno isolato l'uno dall'altro in termini di risorse, sicurezza e errori.

È possibile eseguire i gestori code IBM MQ o le applicazioni IBM MQ nei contenitori.

Informazioni correlate

Cosa sono i contenitori?

OpenShift CP4I Cronologia delle release per IBM MQ Operator

Note:

- Per informazioni sugli operatori IBM MQ precedenti, consultare Release history for IBM MQ Operator nella documentazione IBM MQ 9.3 .
- Per informazioni sui futuri aggiornamenti di IBM MQ , consultare la pagina IBM MQ Correzioni consigliate e date di rilascio della manutenzione pianificata .

IBM MQ Operator 3.6.0

CD

IBM Cloud Pak for Integration Versione

IBM Cloud Pak for Integration 16.1.2

Canale operatore

`v3.6`

Valori consentiti per `.spec.version`

9.4.3.0-r1

Valori consentiti per `.spec.version` durante la migrazione

9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1 ,9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.0.25-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2, 9.4.0.7-

r1, 9.4.0.10-r1, 9.4.0.10-r2, 9.4.0.11-r1, 9.4.0.11-r2, 9.4.0.11-r3, 9.4.1.0-r1, 9.4.1.0-r2, 9.4.1.1-r1, 9.4.2.0-r1, 9.4.2.0-r2, 9.4.2.1-r1, 9.4.2.1-r2

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Novità

- Miglioramento della registrazione della convalida dell'accesso ai webhook, per facilitare la conformità ai requisiti di verifica della registrazione degli accessi. Per ulteriori informazioni, consultare "Configurazione della registrazione di audit per IBM MQ nei container" a pagina 150.
- IBM MQ Operator è stato convalidato per gli ambienti IPv6 dual stack, in linea con i più recenti requisiti di conformità federale degli Stati Uniti.
- Ora è possibile configurare le impostazioni di `ipFamilyPolicy` e `ipFamilies` di IBM MQ queue manager Services. La configurazione è disponibile in `spec.service`. Le impostazioni si applicano a tutti i servizi di gestione delle code.
- È ora possibile configurare i gestori di code distribuiti con IBM MQ Operator con annotazioni di licenza IBM MQ . Per ulteriori informazioni, consultare "Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator" a pagina 160. Si noti che questo aggiornamento si applica a tutti i gestori di code IBM MQ 9.4
- È ora possibile utilizzare le funzionalità di Native HA e Cross-Region Replication sui gestori di code con licenza IBM MQ aggiungendo la nuova licenza IBM MQ Native HA and Cross-Region Replication Add-on. Per ulteriori informazioni, consultare "Configurazione dell'HA nativo e della replica interregionale sui gestori di code con licenza IBM MQ utilizzando il software IBM MQ Operator" a pagina 115. Si noti che questo aggiornamento si applica anche ai gestori di code IBM MQ 9.4.2.

IBM MQ Operator 3.5.3

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.1

Canale operatore
`v3.5`

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

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1 ,9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.0.25-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2, 9.4.0.7-r1, 9.4.0.10-r1, 9.4.0.10-r2, 9.4.0.11-r1, 9.4.1.0-r1, 9.4.1.0-r2, 9.4.1.1-r1, 9.4.2.0-r2, 9.4.2.1-r1

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati

- Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7234178

IBM MQ Operator 3.5.2

CD

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.1

Canale operatore
v3.5

Valori consentiti per `.spec.version`
9.4.2.1-r1

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1 ,9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.0.25-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2, 9.4.0.7-r1, 9.4.0.10-r1, 9.4.0.10-r2, 9.4.0.11-r1, 9.4.1.0-r1, 9.4.1.0-r2, 9.4.1.1-r1, 9.4.2.0-r2

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati

- L'ID e il segreto del client di IBM MQ OpenID Connect (OIDC) sono ora memorizzati in un volume anziché in una variabile d'ambiente.
- Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7232272

IBM MQ Operator 3.5.1

CD

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.1

Canale operatore
v3.5

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

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1 ,9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.0.25-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2, 9.4.0.7-r1, 9.4.0.10-r1, 9.4.0.10-r2, 9.4.1.0-r1, 9.4.1.0-r2, 9.4.1.1-r1, 9.4.2.0-r1

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati

- È stato risolto un problema per cui, in determinate condizioni, la risorsa IntegrationKeycloakClient veniva aggiornata con un ID cliente errato.
- Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7228945

IBM MQ Operator 3.5.0

CD

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.1

Canale operatore
`v3.5`

Valori consentiti per `.spec.version`
9.4.2.0-r1

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.0.25-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2, 9.4.0.7-r1, 9.4.0.10-r1, 9.4.1.0-r1, 9.4.1.0-r2, 9.4.1.1-r1

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Novità

- Quando viene utilizzato un gestore di code, è ora possibile fornire file aggiuntivi. Questi file vengono montati nel contenitore dell' IBM MQ e nelle posizioni specificate. Per ulteriori informazioni, consultare “Esempio: fornitura di file aggiuntivi a un gestore di code” a pagina 102.
- Quando viene utilizzato un gestore di code, è ora possibile fornire ulteriori variabili ambientali. Per ulteriori informazioni, consultare “Esempio: fornitura di variabili ambientali aggiuntive per un gestore di code” a pagina 103.
- IBM MQ i gestori di code di 9.4.2.0-r1 possono essere configurati dall' IBM MQ Operator per fornire metriche tramite HTTPS utilizzando certificati creati da OpenShift. Per ulteriori informazioni, fare riferimento a “Monitoraggio quando si utilizza IBM MQ Operator” a pagina 217.
- IBM MQ CASE 3.5.0 e successivamente contenere solo l'ultima versione del gestore di code. Ad esempio IBM MQ CASE 3.5.0 contiene solo la versione del gestore di code 9.4.2.0-r1. Quando si eseguono installazioni air-gap, questo si traduce in una fase di mirroring dell'immagine più veloce, ma potrebbe essere necessario eseguire ulteriori passaggi di mirroring dell'immagine per installare o aggiornare a un gestore di code che non è l'ultimo. Per ulteriori informazioni, consultare “IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

Elementi modificati
Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7184453

IBM MQ Operator 3.4.1

CD

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.1

Canale operatore
v3.4

Valori consentiti per `.spec.version`
9.4.1.1-r1

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.0.25-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2, 9.4.0.7-r1, 9.4.1.0-r1, 9.4.1.0-r2

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati
Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7182196

IBM MQ Operator 3.4.0

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.1

Canale operatore
v3.4

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

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.0.25-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2, 9.4.0.7-r1, 9.4.1.0-r1

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati

- Un certo numero di operatori vecchi e fuori supporto sono stati rimossi dal CASE.
- Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7178065

IBM MQ Operator 3.3.0

CD

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.3`

Valori consentiti per `.spec.version`
9.4.1.0-r1

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati

- La licenza 'IBM MQ Advanced non può più essere configurata con l'uso di `Sviluppo`.
- I gestori di coda sono ora annotati con le annotazioni di licenza corrette. Per ulteriori informazioni o per istruzioni sull'aggiornamento dei gestori di code precedenti, vedere '“Correzione delle annotazioni di licenza per i gestori di code distribuiti” a pagina 172.
- Le regole relative al valore di '**`spec.license.metric`** sono ora applicate da webhook. I gestori di code esistenti devono seguire queste regole quando vengono aggiornati alla versione 9.4.1.0-r1 e successive.
- Aggiungere il supporto per la licenza " IBM MQ Advanced for Non-Production Environment.
- Rimuovere la possibilità di distribuire con una licenza 'IBM MQ Advanced e '**Use** impostata su `Sviluppo`.
- Ridurre l'ingombro in memoria di 'IBM MQ Operator in cluster di grandi dimensioni.
- Risolvere un problema con le riconciliazioni di '`ServiceAccount` quando si esegue su OCP 4.16.
- Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7174358

IBM MQ Operator 3.2.13

CP4I-SC2

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

Valori consentiti per `.spec.version`
9.4.0.11-r3

Valori consentiti per `.spec.version` durante la migrazione

9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1 ,9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.0.25-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3,

9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2, 9.4.0.7-r1, 9.4.0.10-r1, 9.4.0.10-r2, 9.4.0.11-r1, 9.4.0.11-r2

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

IBM MQ Operator 3.2.12

CP4I-SC2

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

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

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1 ,9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.0.25-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2, 9.4.0.7-r1, 9.4.0.10-r1, 9.4.0.10-r2, 9.4.0.11-r1

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati

- Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7234178

IBM MQ Operator 3.2.11

CP4I-SC2

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

Valori consentiti per `.spec.version`
9.4.0.11-r1

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1 ,9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.0.25-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2, 9.4.0.7-r1, 9.4.0.10-r1, 9.4.0.10-r2

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati

- L'ID e il segreto del client di IBM MQ OpenID Connect (OIDC) sono ora memorizzati in un volume anziché in una variabile d'ambiente.
- Rimosso il segreto di estrazione dell'immagine `sa-<namespace>` , inutilizzato, per evitare gli avvisi di Kubernetes.
- Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7232272

IBM MQ Operator 3.2.10

CP4I-SC2

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

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

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1 ,9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.0.25-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2, 9.4.0.7-r1, 9.4.0.10-r1

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati

- È stato risolto un problema per cui, in determinate condizioni, la risorsa IntegrationKeycloakClient veniva aggiornata con un ID cliente errato.
- Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7228945

IBM MQ Operator 3.2.9

CP4I-SC2

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

Valori consentiti per `.spec.version`
9.4.0.10-r1

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2, 9.4.0.7-r1

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati
Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7184453

IBM MQ Operator 3.2.8

CP4I-SC2

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

Valori consentiti per `.spec.version`
9.4.0.7-r1

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1, 9.4.0.6-r2

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati
Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7182196

IBM MQ Operator 3.2.7

CP4I-SC2

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

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

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1,

9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2, 9.4.0.6-r1

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati
Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7178065

IBM MQ Operator 3.2.6

CP4I-SC2

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

Valori consentiti per `.spec.version`
9.4.0.6-r1

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.0.21-r3, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1, 9.4.0.5-r2

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati
Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7174358

IBM MQ Operator 3.2.5

CD CP4I-SC2

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

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

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.0.21-r2, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2,

9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3, 9.4.0.5-r1

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati
Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7171536

IBM MQ Operator 3.2.4

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

Valori consentiti per `.spec.version`
9.4.0.5-r1

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.0.21-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2, 9.4.0.0-r3

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati
Le vulnerabilità affrontate sono descritte in dettaglio nel presente bollettino di sicurezza: https://www.ibm.com/support/pages/node/7167732

IBM MQ Operator 3.2.3

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

Valori consentiti per `.spec.version`
9.4.0.0-r3

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.0.20-r2, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1, 9.4.0.0-r2

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati
Le vulnerabilità affrontate sono descritte in dettaglio in questi bollettini di sicurezza:

- https://www.ibm.com/support/pages/node/7162095
- https://www.ibm.com/support/pages/node/7162272

IBM MQ Operator 3.2.2

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

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

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3, 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.0.20-r1, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2, 9.4.0.0-r1

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati

- Le vulnerabilità risolte sono descritte in questo Bollettino sulla sicurezza.

IBM MQ Operator 3.2.1

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

Valori consentiti per `.spec.version`
9.4.0.0-r1

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3,9.3.3.2-r3 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.2-r3, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Elementi modificati

- Risolve un problema su OpenShift Container Platform 4.12 in cui l'aggiornamento al Canale v3.2-sc2 potrebbe causare un comportamento imprevisto per gli utenti IBM Cloud Pak for Integration . Per ulteriori informazioni, vedi Aggiornamento da 2023.4 nella documentazione di IBM Cloud Pak for Integration .

IBM MQ Operator 3.2.0

IBM Cloud Pak for Integration Versione
IBM Cloud Pak for Integration 16.1.0

Canale operatore
`v3.2-sc2`

Valori consentiti per `.spec.version`
9.4.0.0-r1

Valori consentiti per `.spec.version` durante la migrazione
9.3.0.0-r1, 9.3.0.0-r2, 9.3.0.0-r3,9.3.3.2-r3 9.3.0.1-r1, 9.3.0.1-r2, 9.3.0.1-r3, 9.3.0.1-r4, 9.3.0.3-r1, 9.3.0.4-r1, 9.3.0.4-r2, 9.3.0.5-r1, 9.3.0.5-r2, 9.3.0.5-r3, 9.3.0.6-r1, 9.3.0.10-r1, 9.3.0.10-r2, 9.3.0.11-r1, 9.3.0.11-r2, 9.3.0.15-r1, 9.3.0.16-r1, 9.3.0.16-r2, 9.3.0.17-r1, 9.3.0.17-r2, 9.3.0.17-r3, 9.3.1.0-r1, 9.3.1.0-r2, 9.3.1.0-r3, 9.3.1.1-r1, 9.3.2.0-r1, 9.3.2.0-r2, 9.3.2.1-r1, 9.3.2.1-r2, 9.3.3.0-r1, 9.3.3.0-r2, 9.3.3.1-r1, 9.3.3.1-r2, 9.3.3.2-r1, 9.3.3.2-r2, 9.3.3.2-r3, 9.3.3.3-r1, 9.3.3.3-r2, 9.3.4.0-r1, 9.3.4.1-r1, 9.3.5.0-r1, 9.3.5.0-r2, 9.3.5.1-r1, 9.3.5.1-r2

Red Hat OpenShift Container Platform versioni
OpenShift Container Platform 4.12 e successive.

IBM Cloud Pak foundational services versioni
Solo IBM Cloud Pak foundational services versione 4.6 .

Novità

- “Espansione dei volumi persistenti” a pagina 167 è ora supportata.
- I gestori code possono ora essere arrestati aggiungendo l'annotazione `mq.ibm.com/stop` e impostandola su `true`. Vedi“Arresto di un gestore code (mq.ibm.com/stop)” a pagina 228

  Note:

  – Un gestore code arrestato ha il campo *.replicas* nel relativo `StatefulSet` impostato su 0.
  – Poiché IBM MQ Operator ora gestisce attivamente il campo *.replicas* in `StatefulSet`, se si modifica questo campo viene immediatamente ripristinato dall'operatore.
  – Le versioni precedenti di IBM MQ immettono uno stato 'Non riuscito' se si modifica il campo *.replicas* , ma si conserva ancora il valore modificato. Se le tue procedure operative esistenti si basano su questo comportamento, da IBM MQ 9.4 devi utilizzare l'annotazione `mq.ibm.com/stop` .

Elementi modificati

- Sono ora supportate le release con numero dispari di Red Hat OpenShift Container Platform .
- IBM MQ L'immagine di catalogo è stata spostata nel formato di catalogo basato su file dal formato database SQLite .
- Basato su Red Hat Immagine base universale 9.4-949.1716471857. **Nota:** UBI 9 è in attesa di certificazione FIPS 140-3 . L'UBI 9 non è supportato dall'architettura Power 8.
- Le vulnerabilità risolte sono descritte in questo Bollettino sulla sicurezza.

OpenShift CP4I Cronologia dei rilasci per le immagini dei container di queue manager

Nota: Per informazioni sulle immagini precedenti del contenitore del gestore di code, vedere la cronologia dei rilasci per 'IBM MQ Operator nella documentazione di 'IBM MQ 9.3.

Immagini supportate da sole o con il " IBM MQ Operator

9.4.3.0-r1

CD

Versione operatore compatibile
Supportato da solo o con IBM MQ Operator 3.6.0 o superiore.

Architetture supportate
`amd64`, `s390x`, `ppc64le`

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.3.0-r1` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.3.0-r1` (IBM MQ Advanced)

  Nota: È inoltre possibile utilizzare questa immagine per distribuire i gestori di code con licenza IBM MQ , se si rispettano i termini di licenza appropriati e si annotano questi gestori di code. Per ulteriori informazioni, consultare “Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator” a pagina 160 e “IBM MQ annotazioni sulla licenza del contenitore” a pagina 281.
- `icr.io/ibm-messaging/mq:9.4.3.0-r1` (IBM MQ Advanced for Developers)

Novità

- È ora possibile abilitare la registrazione degli accessi agli endpoint delle metriche di Prometheus. Per ulteriori informazioni, consultare “Configurazione della registrazione di audit per IBM MQ nei container” a pagina 150.
- È ora possibile configurare i gestori di code distribuiti con IBM MQ Operator con annotazioni di licenza IBM MQ . Per ulteriori informazioni, consultare “Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator” a pagina 160. Si noti che questo aggiornamento si applica a tutti i gestori di code IBM MQ 9.4
- È ora possibile utilizzare le funzionalità di Native HA e Cross-Region Replication sui gestori di code con licenza IBM MQ aggiungendo la nuova licenza IBM MQ Native HA and Cross-Region Replication Add-on. Per ulteriori informazioni, vedere “Configurazione dell'HA nativo e della replica interregionale sui gestori di code con licenza IBM MQ utilizzando il software IBM MQ Operator” a pagina 115 o “Configurazione di Native HA e Cross-Region Replication su gestori di code con licenza IBM MQ in Kubernetes” a pagina 186. Si noti che questo aggiornamento si applica anche ai gestori di code IBM MQ 9.4.2.

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.3 .
- Le vulnerabilità affrontate sono descritte in dettaglio in questo bollettino di sicurezza: https://www.ibm.com/support/pages/node/7236608

9.4.2.1-r2

CD

Versione operatore compatibile
Supportato da solo o con IBM MQ Operator 3.5.3 o superiore.

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.2.1-r2` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.2.1-r2` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.2.1-r2` (IBM MQ Advanced for Developers)

Novità

- Cosa c'è di nuovo in 'IBM MQ 9.4.2 per Multiplatforms - base e Advanced entitlement.

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.2 .
- Le vulnerabilità affrontate sono descritte in dettaglio in questo bollettino di sicurezza: https://www.ibm.com/support/pages/node/7234178

9.4.2.1-r1

CD

Versione operatore compatibile
Supportato da solo o con IBM MQ Operator 3.5.2 o superiore.

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.2.1-r1` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.2.1-r1` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.2.1-r1` (IBM MQ Advanced for Developers)

Novità

- Cosa c'è di nuovo in 'IBM MQ 9.4.2 per Multiplatforms - base e Advanced entitlement.

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.2 .
- Le vulnerabilità affrontate sono descritte in dettaglio in questo bollettino di sicurezza: https://www.ibm.com/support/pages/node/7232272

9.4.2.0-r2

CD

Versione operatore compatibile
Supportato da solo o con IBM MQ Operator 3.5.1 o superiore.

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.2.0-r2` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.2.0-r2` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.2.0-r2` (IBM MQ Advanced for Developers)

Novità

- Cosa c'è di nuovo in 'IBM MQ 9.4.2 per Multiplatforms - base e Advanced entitlement.

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.2 .
- Le vulnerabilità affrontate sono descritte in dettaglio in questo bollettino di sicurezza: https://www.ibm.com/support/pages/node/7228945

9.4.2.0-r1

CD

Versione operatore compatibile
Supportato in modalità standalone o con un IBM MQ Operator 3.5.0 o superiore.

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.2.0-r1` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.2.0-r1` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.2.0-r1` (IBM MQ Advanced for Developers)

Novità

- Le metriche di Queue Manager possono essere servite tramite HTTPS popolando `/etc/mqm/metrics/pki/keys` con chiavi TLS. Per ulteriori informazioni, consultare “Sistema di file e punti di montaggio” a pagina 290.
- Cosa c'è di nuovo in 'IBM MQ 9.4.2 per Multiplatforms - base e Advanced entitlement.

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.2 .
- Le vulnerabilità affrontate sono descritte in dettaglio in questo bollettino di sicurezza: https://www.ibm.com/support/pages/node/7184453

9.4.1.1-r1

CD

Versione operatore compatibile
Supportato da solo o con IBM MQ Operator 3.4.1 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.1.1-r1` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.1.1-r1` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.1.1-r1` (IBM MQ Advanced for Developers)

Novità

- Cosa c'è di nuovo in 'IBM MQ 9.4.1 per Multiplatforms - base e Advanced entitlement.

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.1 .

- Le vulnerabilità affrontate sono descritte in dettaglio in questo bollettino di sicurezza: https://www.ibm.com/support/pages/node/7182196

9.4.1.0-r2

CD

Versione operatore compatibile
Supportato da solo o con 'IBM MQ Operator 3 3.4.0 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.1.0-r2` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.1.0-r2` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.1.0-r2` (IBM MQ Advanced for Developers)

Novità

- Cosa c'è di nuovo in 'IBM MQ 9.4.1 per Multiplatforms - base e Advanced entitlement.

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.1 .
- Le vulnerabilità affrontate sono descritte in dettaglio in questo bollettino di sicurezza: https://www.ibm.com/support/pages/node/7178065

9.4.1.0-r1

CD

Versione operatore compatibile
Supportato standalone o con 'IBM MQ Operator 3 3.3.0 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.1.0-r1` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.1.0-r1` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.1.0-r1` (IBM MQ Advanced for Developers)

Novità

- La licenza 'IBM MQ Advanced for Non-Production Environment è ora configurabile per i gestori di code 9.4.1.0-r1 e successivi. Per informazioni sulla licenza, vedere '“Licenze e riferimenti alle API per IBM MQ Operator” a pagina 248.
- Novità in IBM MQ 9.4.1 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Correzione per APAR IT46430.
- Correzione per risolvere un problema di elaborazione dei certificati intermedi.
- Correzione per disabilitare la modalità FIPS per la generazione del deposito di chiavi **runmqakm** , quando FIPS non è abilitato.
- Correzione per sanificare i percorsi dei file.
- Aggiornamenti per ridurre le dimensioni dell'immagine.

- l'immagine del contenitore del gestore di code IBM MQ 9.4.1.0-r1 include la versione 3.1.112024.3.0) dell'uscita utente di tracciamento 'IBM MQ. Vedere il tracciamento di 'IBM MQ nella documentazione di 'IBM Instana.
- Consentire che la configurazione di Native HA sia fornita esternamente anziché generata dal modello.
  - Deprecare l'uso della configurazione delle variabili d'ambiente di Native HA (tranne 'MQ_NATIVE_HA=true, che è ancora richiesto).
  - Chiarire il comportamento della configurazione della variabile d'ambiente, ora deprecata, in 'IBM Documentation.
- Cosa è cambiato inIBM MQ 9.4.1 .
- Le vulnerabilità affrontate sono descritte in dettaglio in questo bollettino di sicurezza: https://www.ibm.com/support/pages/node/7174358

Immagini supportate solo per l'uso con il " IBM MQ Operator

9.4.0.11-r3

CP4I-SC2

Versione operatore richiesta
"IBM MQ Operator 3.2.13" a pagina 10 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.11-r3` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.11-r3` (IBM MQ Advanced)

  Nota: È inoltre possibile utilizzare questa immagine per distribuire i gestori di code con licenza IBM MQ , se si rispettano i termini di licenza appropriati e si annotano questi gestori di code. Per ulteriori informazioni, consultare "Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator" a pagina 160 e "IBM MQ annotazioni sulla licenza del contenitore" a pagina 281.
- `icr.io/ibm-messaging/mq:9.4.0.11-r3` (IBM MQ Advanced for Developers)

Novità

- È ora possibile configurare i gestori di code distribuiti con IBM MQ Operator con annotazioni di licenza IBM MQ . Per ulteriori informazioni, consultare "Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator" a pagina 160. Si noti che questo aggiornamento si applica a tutti i gestori di code IBM MQ 9.4

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.0 .
- Le vulnerabilità che vengono affrontate sono descritte in dettaglio in questo bollettino di sicurezza: https://www.ibm.com/support/pages/node/7236608

9.4.0.11-r2

CP4I-SC2

Versione operatore richiesta
"IBM MQ Operator 3.2.12" a pagina 11 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.11-r2` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.11-r2` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.0.11-r2` (IBM MQ Advanced for Developers)

Novità

- Novità in IBM MQ 9.4.0 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.0 .
- Le vulnerabilità che vengono affrontate sono descritte in dettaglio in questo bollettino di sicurezza: https://www.ibm.com/support/pages/node/7234178

9.4.0.11-r1

CP4I-SC2

Versione operatore richiesta
3.2.11 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.11-r1` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.11-r1` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.0.11-r1` (IBM MQ Advanced for Developers)

Novità

- Novità in IBM MQ 9.4.0 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.0 .

9.4.0.10-r2

CP4I-SC2

Versione operatore richiesta
3.2.10 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.10-r2` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.10-r2` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.0.10-r2` (IBM MQ Advanced for Developers)

Novità

- Novità in IBM MQ 9.4.0 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.0 .

9.4.0.10-r1

CP4I-SC2

Versione operatore richiesta
3.2.9 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.10-r1` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.10-r1` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.0.10-r1` (IBM MQ Advanced for Developers)

Novità

- Novità in IBM MQ 9.4.0 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.0 .

9.4.0.7-r1

CP4I-SC2

Versione operatore richiesta
3.2.8 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.7-r1` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.7-r1` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.0.7-r1` (IBM MQ Advanced for Developers)

Novità

- Novità in IBM MQ 9.4.0 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.0 .

9.4.0.6-r2

CP4I-SC2

Versione operatore richiesta
3.3.2.7 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.6-r2` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.6-r2` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.0.6-r2` (IBM MQ Advanced for Developers)

Novità

- Novità in IBM MQ 9.4.0 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.0 .

9.4.0.6-r1

CP4I-SC2

Versione operatore richiesta
3.3.2.6 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.6-r1` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.6-r1` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.0.6-r1` (IBM MQ Advanced for Developers)

Novità

- Novità in IBM MQ 9.4.0 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.0 .
- Applicata una correzione per risolvere il problema del blocco del container durante l'esecuzione del comando '**endmqm** su Mac con Rosetta.

9.4.0.5-r2

CD CP4I-SC2

Versione operatore richiesta
3.2.5 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.5-r2` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.5-r2` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.0.5-r2` (IBM MQ Advanced for Developers)

Novità

- Novità in IBM MQ 9.4.0 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.0 .
- Basato su Red Hat Universal Base Image 9.4-1227.1725849298.

9.4.0.5-r1

CD CP4I-SC2

Versione operatore richiesta
3.2.4 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.5-r1` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.5-r1` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.0.5-r1` (IBM MQ Advanced for Developers)

Novità

- Novità in IBM MQ 9.4.0 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.0 .
- Basato su Red Hat Universal Base Image 9.4-1194.

9.4.0.0-r3

Versione operatore richiesta
3.2.3 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.0-r3` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.0-r3` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.0.0-r3` (IBM MQ Advanced for Developers)

Novità

- Novità in IBM MQ 9.4.0 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.0 .
- Basato suRed Hat Immagine di base universale9.4-1134 .
- Risolve un problema riscontrato solo su OpenShift Container Platform (OCP) 4.16, dove la riconciliazione `ServiceAccount` crea un ciclo indefinito di segreti di estrazione delle immagini. OCP 4.16 ha introdotto un cambiamento di comportamento per cui OCP inietta `.metadata.annotation` nel `ServiceAccount`.

9.4.0.0-r2

Versione operatore richiesta
3.2.2 o più alto

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.0-r2` (IBM Cloud Pak for Integration)
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.0-r2` (IBM MQ Advanced)
- `icr.io/ibm-messaging/mq:9.4.0.0-r2` (IBM MQ Advanced for Developers)

Novità

- Novità in IBM MQ 9.4.0 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Cosa è cambiato inIBM MQ 9.4.0 .
- Basato suRed Hat Immagine di base universale9.4-1134 .

9.4.0.0-r1

CD CP4I-SC2

Versione operatore richiesta
3.2.0 o superiore

Architetture supportate
amd64, s390x, ppc64le

Immagini

- `cp.icr.io/cp/ibm-mqadvanced-server-integration:9.4.0.0-r1 (IBM Cloud Pak for Integration)`
- `cp.icr.io/cp/ibm-mqadvanced-server:9.4.0.0-r1 (IBM MQ Advanced)`
- `icr.io/ibm-messaging/mq:9.4.0.0-r1 (IBM MQ Advanced for Developers)`

Novità

- Novità in IBM MQ 9.4.0 for Multiplatforms - base e titolarità avanzata

Elementi modificati

- Cosa è cambiato in IBM MQ 9.4.0
- Deprecated Quando utilizzi IBM MQ Advanced for Developers, l'impostazione delle parole d'ordine per gli utenti `admin` e `app` tramite una variabile di ambiente è obsoleta. Utilizzare invece i segreti.
- È stato aggiunto un nuovo valore facoltativo `mqsc` per la variabile di ambiente *MQ_LOGGING_CONSOLE_SOURCE*. Questa opzione può essere utilizzata per riflettere i contenuti di `autocfgmqsc.LOG` nel log del contenitore.
- Basato su Red Hat Immagine base universale 9.4-949.1716471857. **Nota:** UBI 9 è in attesa di certificazione FIPS 140-3 . L'UBI 9 non è supportato dall'architettura Power 8.

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 OpenShift Container Platform architecture).

Procedura

- “Scelta della modalità di utilizzo di IBM MQ nei contenitori” a pagina 28.
- “Supporto per IBM MQ nei contenitori” a pagina 28.
- “Pianificazione dell'archiviazione per IBM MQ Operator” a pagina 38.
- “Pianificazione dell'alta disponibilità per IBM MQ nei contenitori” a pagina 40.
- “Ripristino di emergenza per IBM MQ nei contenitori” a pagina 46.

- “Autenticazione e autorizzazione utente per IBM MQ nei contenitori” a pagina 49.

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

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

IBM MQ Operator estende l'API Red Hat OpenShift Container Platform per aggiungere una nuova risorsa personalizzata `QueueManager` . 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. Vedere “Considerazioni sull'esecuzione di un aggiornamento continuo di un gestore code HA nativo” a pagina 44

Alcune funzioni IBM MQ non sono supportate quando si utilizza IBM MQ Operator. Vedere “Supporto per l'opzione IBM MQ Operator” a pagina 29 per i dettagli su ciò che è supportato quando si utilizza IBM MQ Operator.

Utilizzo delle immagini precostituite IBM MQ Advanced

È possibile utilizzare le immagini precostituite disponibili nel file IBM Container Registry. Per ulteriori dettagli vedere “Supporto per le immagini del contenitore IBM MQ Advanced” a pagina 30.

Utilizzo delle immagini precostituite con licenza IBM MQ

Da IBM MQ 9.4.1, è possibile utilizzare le immagini precostituite disponibili da IBM Container Registry. Per ulteriori informazioni, consultare “Supporto per le immagini dei container con licenza IBM MQ ” a pagina 31.

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.

Sono disponibili esempi per la creazione di immagini personalizzate. Consultare “Preparazione per IBM MQ creando la propria immagine contenitore” a pagina 88.

Vedere “Supporto per la creazione di immagini container IBM MQ personalizzate” a pagina 33 per i dettagli su ciò che è supportato quando si costruisce la propria immagine e il codice di distribuzione.

Riferimenti correlati

“Supporto per IBM MQ nei contenitori” a pagina 28
Non tutte le funzioni IBM MQ sono disponibili e supportate nello stesso modo nei contenitori.

Supporto per IBM MQ nei contenitori

Non tutte le funzioni IBM MQ sono disponibili e supportate nello stesso modo nei contenitori.

Concetti correlati

Domande frequenti di IBM MQ per le release Long Term Support e Continuous Delivery

Riferimenti correlati

IBM Cloud Pak for Integration Software Support Lifecycle Addendum

Informazioni correlate

Supporto IBM MQ per SELinux e AppArmor

Supporto per l'opzione IBM MQ Operator

L'elemento IBM MQ Operator è supportato come parte di IBM MQ Advanced e IBM Cloud Pak for Integration

Piattaforme supportate

L'opzione IBM MQ Operator è supportata solo da Red Hat OpenShift Container Platform. I rilasci di Red Hat OpenShift Container Platform non sono più supportati da IBM MQ una volta che Red Hat cessa il supporto. Vedere “Supporto versione per IBM MQ Operator” a pagina 35 e Red Hat OpenShift Container Platform Date del ciclo di vita per maggiori dettagli

Architetture CPU supportate

L'opzione IBM MQ Operator è supportata su `amd64` e `s390x` z/Linux, e su `ppc64le` Power Systems versione 9 e superiori. Si noti che tutti i nodi del cluster Red Hat OpenShift Container Platform devono utilizzare la stessa architettura di CPU.

Durata del sostegno

IBM fornisce il supporto per i difetti e l'utilizzo di IBM MQ Operator e delle immagini di queue manager associate. La durata dell'assistenza dipende dalla licenza utilizzata:

- Se si utilizza una licenza **IBM Cloud Pak for Integration**, la versione IBM MQ Operator è supportata per la stessa durata della corrispondente versione IBM Cloud Pak for Integration. Alcune versioni sono IBM Cloud Pak for Integration - Support Cycle 2 (formerly Long Term Support) e altre sono Continuous Delivery (CD). Il gestore CD e i gestori di code sono supportati fino alla prossima release IBM Cloud Pak for Integration CD o CP4I-LTS L'operatore CP4I-LTS e i gestori di code distribuiti con la licenza IBM Cloud Pak for Integration sono supportati fino alla prossima release CP4I-LTS, più un periodo di grazia per consentire l'aggiornamento.

  Ad esempio, se si crea un gestore di code utilizzando la licenza IBM Cloud Pak for Integration “L-JTPV-KYG8TF”, questa è associata a una release CP4I-LTS. Questo distribuirà IBM MQ Advanced 9.4.0, e sarà supportato in linea con quella versione IBM Cloud Pak for Integration (due anni).
- Se si utilizza un elemento **IBM MQ Advanced** o una licenza **IBM MQ** è supportato solo il flusso IBM MQ Continuous Delivery . Ogni IBM MQ Operator e queue manager distribuito con una licenza IBM MQ Advanced o IBM MQ è supportato per un anno dal primo operatore che supporta quella versione di queue manager. I rilasci ricevono proattivamente le correzioni di sicurezza fino al successivo rilascio Continuous Delivery . Ad esempio, se si crea un gestore di code utilizzando la licenza IBM MQ Advanced “L-EHXT-MQCRN9”, esso utilizzerà IBM MQ Advanced 9.4.0 come release Continuous Delivery e sarà supportato per un anno, mentre le correzioni di sicurezza saranno pubblicate fino al rilascio di IBM MQ 9.4.1. Dopo il rilascio della versione 9.4.1, su richiesta IBM valuterà la possibilità di compiere sforzi ragionevoli per fornire una correzione sul livello di rilascio del CD precedente

Per ulteriori informazioni, consultare le IBM MQ FAQ per il supporto a lungo termine e le release di consegna continua.

Fissare la disponibilità

Le correzioni periodiche e provvisorie del gestore delle code sono disponibili presso IBM Container Registry. Non sono disponibili correzioni provvisorie per il IBM MQ Operator.

Caratteristiche non disponibili

Le seguenti caratteristiche di IBM MQ Advanced non sono disponibili per i gestori di code gestiti da IBM MQ Operator:

- I gestori di code di dati replicati (RDQM) non sono disponibili: questa funzione è strettamente legata al kernel Linux, quindi non è supportata nei container
- La registrazione lineare per i gestori di code non è disponibile
- Le opzioni personalizzate della riga di comando per **crtmqdir**, **crtmqm**, **strmqm** e **endmqm** non sono disponibili - La maggior parte delle opzioni può essere configurata utilizzando un file INI, ma alcune non possono essere configurate, come l'uso della registrazione lineare.
- Gli utenti del sistema operativo (OS) non sono disponibili
- AMQP non è installato
- MQTT non è installato
- L'agente di trasferimento file gestito non è disponibile. Tuttavia, è possibile utilizzare l'operatore IBM MQ per fornire uno o più gestori di code di coordinamento, comando o agente.
- La sicurezza avanzata dei messaggi con intercettazione MCA tramite un archivio chiavi separato non è disponibile. L'intercettazione MCA può ancora essere configurata per utilizzare l'archivio chiavi del gestore di code principale.

È possibile utilizzare solo le immagini del contenitore del gestore di code fornite da IBM con l'Operatore MQ e non è possibile creare una propria immagine con codice personalizzato. L'eccezione è rappresentata dall'aggiunta di un livello di immagine contenitore a un'immagine IBM fornita, per aggiungere file di configurazione o uscite.

Concetti correlati

Domande frequenti di IBM MQ per le release Long Term Support e Continuous Delivery

Riferimenti correlati

IBM Cloud Pak for Integration Addendum al ciclo di vita del supporto software

Informazioni correlate

Date del ciclo di vita di IBM MQ nei container

Supporto per le immagini del contenitore IBM MQ Advanced

A partire dalla versione IBM MQ Advanced 9.4.1, è possibile utilizzare l'immagine del contenitore precostituito IBM MQ Advanced senza l'opzione IBM MQ Operator

L'immagine contenitore precostituita IBM MQ Advanced è composta da diversi livelli logici:

- IBM MQ codice di abilitazione del contenitore (`runmqserver`, `chkmqhealthy`, `chkmqready` e `chkmqstarted`)
- IBM MQ Advanced stesso
- Red Hat pacchetti dal Red Hat Universal Base Image

Piattaforme supportate

L'immagine precostituita IBM MQ Advanced può essere distribuita negli ambienti di runtime dei container forniti da containerd, cri-o, Podman o Docker. Per maggiori dettagli, vedere Requisiti di sistema per IBM MQ. IBM fornisce supporto per i difetti e l'utilizzo del codice di abilitazione del contenitore MQ IBM e di 'IBM MQ Advanced stesso, ma non aiuta gli utenti a configurare ed eseguire il debug dei problemi con il loro ambiente di contenitore o la pipeline di distribuzione.

Importante: Red Hat fornisce il supporto per i pacchetti 'Red Hat solo per i contenitori in esecuzione su host 'Red Hat Enterprise Linux® o 'Red Hat Enterprise Linux CoreOS. Per ulteriori informazioni, consultare la Politica di supporto per i container diRed Hat.

Licenze

L'immagine precostituita IBM MQ Advanced è idonea per la concessione di licenze per container se utilizzata su Kubernetes con il IBM License Service. Se non si utilizza l'orchestrazione Kubernetes (ad esempio, utilizzando Podman o Docker), non è possibile sfruttare questo modello di licenza ottimizzato. Per maggiori dettagli, vedere Frequenze sulle licenze dei container.

Architetture CPU supportate

L'immagine precostituita IBM MQ Advanced è supportata su `amd64` e `s390x` z/Linux, e su `ppc64le` Power Systems versione 9 e superiori. Si noti che tutti i contenitori che fanno parte dello stesso gestore di code (ad esempio, le repliche Native HA) devono utilizzare la stessa architettura di CPU.

Durata del sostegno

È supportato solo il flusso IBM MQ Advanced Continuous Delivery Ogni versione di queue manager è supportata secondo la normale durata del supporto (un anno).

I rilasci ricevono proattivamente le correzioni di sicurezza fino al rilascio successivo Continuous Delivery. Per ulteriori informazioni, consultate IBM MQ FAQ per il supporto a lungo termine e le release di consegna continua.

Fissare la disponibilità

Le correzioni periodiche e intermedie sono disponibili presso IBM Container Registry.

Caratteristiche non disponibili

Le seguenti caratteristiche di IBM MQ Advanced non sono disponibili con l'immagine del contenitore precostruita:

- I gestori di code di dati replicati (RDQM) non sono disponibili: questa funzione è strettamente legata al kernel Linux, quindi non è supportata nei container
- La registrazione lineare per i gestori di code non è disponibile
- Le opzioni personalizzate della riga di comando per **`crtmqdir`**, **`crtmqm`**, **`strmqm`** e **`endmqm`** non sono disponibili - La maggior parte delle opzioni può essere configurata utilizzando un file INI, ma alcune non possono essere configurate, come l'uso della registrazione lineare.
- Gli utenti del sistema operativo (OS) non sono disponibili
- AMQP non è installato
- MQTT non è installato
- L'agente di trasferimento file gestito non è disponibile. Tuttavia, è possibile utilizzare l'immagine del contenitore precostituito per fornire uno o più gestori di code di coordinamento, comando o agente.

Se si desidera utilizzare queste funzionalità, è possibile creare la propria immagine del contenitore.

Concetti correlati

Domande frequenti di IBM MQ per le release Long Term Support e Continuous Delivery

Red Hat Politica di supporto per i container

Informazioni correlate

Date del ciclo di vita di IBM MQ nei container

V 9.4.1 Supporto per le immagini dei container con licenza IBM MQ

A partire da IBM MQ 9.4.1 è possibile utilizzare l'immagine del contenitore precostruita con licenza IBM MQ senza IBM MQ Operator.

L'immagine precostituita del contenitore IBM MQ è composta da diversi livelli logici:

- IBM MQ codice di abilitazione del contenitore (`runmqserver`, `chkmqhealthy`, `chkmqready` e `chkmqstarted`)

- IBM MQ stesso
- Red Hat dal pacchetto Red Hat Universal Base Image

Nota: L'immagine del contenitore con licenza IBM MQ utilizza l'immagine IBM MQ Advanced , anche se le funzionalità specifiche di IBM MQ Advanced non possono essere utilizzate senza le licenze appropriate. Per ulteriori informazioni, consultare la sezione “Caratteristiche non disponibili da IBM MQ Advanced” a pagina 32 più avanti in questo argomento.

Piattaforme supportate

L'immagine precostituita di IBM MQ può essere distribuita negli ambienti di runtime dei container forniti da containerd, cri-o, Podman o Docker. Per maggiori dettagli, vedere Requisiti di sistema per IBM MQ. IBM fornisce supporto per i difetti e l'uso del codice di abilitazione del contenitore IBM fornito da IBM MQ e di IBM MQ stesso, ma non aiuta gli utenti a configurare ed eseguire il debug dei problemi con il loro ambiente di contenitore o la pipeline di distribuzione.

Importante: Red Hat fornisce il supporto per i pacchetti Red Hat solo per i contenitori in esecuzione su host Red Hat Enterprise Linux o Red Hat Enterprise Linux CoreOS . Per ulteriori informazioni, vedere Red Hat Container Support Policy.

Licenze

L'immagine precostruita IBM MQ è idonea per le licenze per container se utilizzata su Kubernetes con l'immagine IBM License Service. Se non si utilizza l'orchestrazione Kubernetes (ad esempio, utilizzando Podman o Docker), non è possibile sfruttare questo modello di licenza ottimizzato. Per maggiori dettagli, consultare le FAQ sulle licenze per i container.

Architetture CPU supportate

L'immagine precostituita di IBM MQ è supportata su `amd64` e `s390x` z/ Linux, e su `ppc64le` Power Systems versione 9 e successive. Si noti che tutti i contenitori che fanno parte dello stesso gestore di code (ad esempio, le repliche Native HA) devono utilizzare la stessa architettura di CPU.

Durata del sostegno

È supportato solo il flusso IBM MQ Continuous Delivery. Ogni versione di queue manager è supportata secondo la normale durata del supporto (un anno). I rilasci ricevono proattivamente le correzioni di sicurezza fino al successivo rilascio Continuous Delivery . Per ulteriori informazioni, consultate le FAQ di IBM MQ per il supporto a lungo termine e le release di Continuous Delivery.

Fissare la disponibilità

Le correzioni periodiche e intermedie sono disponibili su IBM Container Registry.

Caratteristiche non disponibili da IBM MQ Advanced

Le seguenti funzioni di IBM MQ Advanced non possono essere utilizzate con l'immagine del contenitore precostruita con licenza IBM MQ :

- HA nativo e replica interregionale. Per utilizzare le funzionalità Native HA e Cross-Region Replication su un gestore di code con licenza IBM MQ , vedere “Configurazione dell'HA nativo e della replica interregionale sui gestori di code con licenza IBM MQ utilizzando il software IBM MQ Operator” a pagina 115.
- AMS
- Managed File Transfer supporto per il tipo OpenSSH Chiave privata per i server SFTP
- Autorecupero degli oggetti danneggiati per i gestori di code Native HA
- Pianificazione delle immagini multimediali - registrazione replicata
- RDQM disponibile su Red Hat Enterprise Linux 9

- Kafka Connetti

Se si desidera utilizzare queste funzionalità, è possibile distribuire IBM MQ Advanced. Per ulteriori informazioni, vedere Supporto per le immagini del contenitore IBM MQ Advanced.

Linux Supporto per la creazione di immagini container IBM MQ personalizzate

L'esempio del contenitore 'IBM MQ fornisce un codice wrapper non supportato per abilitare 'IBM MQ in un contenitore. Questa è la soluzione più flessibile per i container, ma richiede forti competenze nella configurazione dei container e la "proprietà" del container risultante.

Download

L'esempio del contenitore 'IBM MQ è disponibile qui: https://github.com/ibm-messaging/mq-container.

Nota: L'esempio del contenitore, e qualsiasi codice aggiunto o modificato, non è supportato da 'IBM.

Per l'uso del contenitore di esempio, si scarica un'immagine 'IBM MQ non installata.

- Se si dispone di una licenza 'IBM MQ Advanced for Developers, utilizzare la versione 'Continuous Delivery (CD) di 'IBM MQ. Queste immagini non installate sono disponibili nel CD Downloading IBM MQ 9.4.
- Se si dispone di una licenza Production, utilizzare la versione 'Long Term Support (LTS) o 'CD di 'IBM MQ. Queste immagini non installate sono disponibili presso Fix Central.

Piattaforme supportate

Vedere Requisiti di sistema per IBM MQ per informazioni dettagliate sugli ambienti di runtime dei container e sulle distribuzioni Linux supportate. IBM MQ può essere utilizzato con gli ambienti di runtime dei container forniti da containerd, cri-o, Podman o 'Docker. IBM fornisce supporto per i difetti e l'utilizzo del software di base 'IBM MQ, ma non aiuta a configurare ed eseguire il debug dei problemi con l'ambiente del container o la pipeline di distribuzione.

L'esempio del contenitore utilizza un Red Hat Universal Base Image come immagine di base. Red Hat fornisce il supporto per i pacchetti Red Hat solo per i container in esecuzione su host Red Hat Enterprise Linux o Red Hat Enterprise Linux CoreOS. Per ulteriori informazioni, consultare la Politica di supporto per i container diRed Hat. È possibile creare la propria immagine contenitore su qualsiasi immagine di base 'Linux che corrisponda a una distribuzione 'Linux supportata da 'IBM MQ, ad esempio:

- Red Hat Universal Base Image
- Ubuntu
- SUSE Linux Enterprise Server Immagine del contenitore di base

Architetture CPU supportate

Vedere Requisiti di sistema per 'IBM MQ.

Durata del supporto per " IBM MQ

Il supporto è fornito per le versioni di produzione delle immagini non installate " IBM MQ. Per il 'LTS, la durata del supporto è di 5 anni dalla disponibilità generale. Per il 'CD, ogni release incrementale è supportata per 1 anno dalla disponibilità generale. Per ulteriori informazioni, vedere il documento 'IBM 'MQ FAQ per il supporto a lungo termine e i rilasci in consegna continua.

Fissare la disponibilità per 'IBM MQ

Per le versioni di produzione delle immagini non installate " IBM MQ, le correzioni vengono fornite per tutta la durata del supporto. È inoltre possibile richiedere correzioni. Se un aggiornamento 'CD non è più

idoneo per la correzione dei difetti, aggiornate a una versione idonea e, se il difetto può essere ricreato, verrà fornita una correzione. Per informazioni sulle release 'LTS e 'CD disponibili e sulle ultime correzioni, vedere Correzioni diIBM MQ per versione.

Caratteristiche non disponibili

Non è possibile utilizzare utenti del sistema operativo (OS) con un'immagine non installata " IBM MQ. Se si desidera utilizzare gli utenti del sistema operativo, è necessario creare la propria immagine utilizzando il programma di installazione RPM, per il quale non esiste alcun esempio.

Per tutti i dettagli sui componenti supportati, vedere i requisiti di sistema per 'IBM MQ. Sono disponibili alcuni campioni, ma non per tutti i componenti.

Concetti correlati

Domande frequenti di IBM MQ per le release Long Term Support e Continuous Delivery

Attività correlate

“Preparazione per IBM MQ creando la propria immagine contenitore” a pagina 88

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.

MQ Adv. Linux Disponibilità dell'immagine del contenitore IBM MQ Advanced for Developers

L'immagine precostituita IBM MQ Advanced for Developers è disponibile presso IBM Container Registry: `icr.io/ibm-messaging/mq`. L'immagine è un download gratuito e non è giustificata. **L'immagine non è supportata.**

Piattaforme disponibili

L'immagine precostituita IBM MQ Advanced for Developers può essere distribuita negli ambienti di runtime dei container forniti da containerd, cri-o, Podman o Docker. Per ulteriori dettagli, vedere Requisiti di sistema per IBM MQ.

Architetture CPU disponibili

L'immagine precostituita IBM MQ Advanced for Developers è disponibile su `amd64` e `s390x` z/Linux, e su `ppc64le` Power Systems versione 9 e superiori. Si noti che tutti i contenitori che fanno parte dello stesso gestore di code (ad esempio, le repliche Native HA) devono utilizzare la stessa architettura di CPU.

Caratteristiche non disponibili

Le seguenti caratteristiche di IBM MQ Advanced for Developers non sono disponibili con l'immagine del contenitore precostruita:

- I gestori di code di dati replicati (RDQM) non sono disponibili: questa funzione è strettamente legata al kernel Linux, quindi non è supportata nei container
- La registrazione lineare per i gestori di code non è disponibile
- Le opzioni personalizzate della riga di comando per **`crtmqdir`**, **`crtmqm`**, **`strmqm`** e **`endmqm`** non sono disponibili - La maggior parte delle opzioni può essere configurata utilizzando un file INI, ma alcune non possono essere configurate, come l'uso della registrazione lineare.
- Gli utenti del sistema operativo (OS) non sono disponibili
- AMQP non è installato
- MQTT non è installato
- L'agente di trasferimento file gestito non è disponibile. Tuttavia, è possibile utilizzare l'immagine del contenitore precostituito per fornire uno o più gestori di code di coordinamento, comando o agente.

Concetti correlati

Domande frequenti di IBM MQ per le release Long Term Support e Continuous Delivery

“IBM MQ Advanced for Developers immagine contenitore” a pagina 291

Un'immagine del contenitore precostruita è disponibile per IBM MQ Advanced for Developers. Questa immagine è disponibile da IBM Container Registry. Questa immagine è adatta all'uso con Docker, Podman, Kubernetes e altri ambienti container.

OpenShift CP4I CD CP4I-SC2 Supporto versione per IBM MQ Operator

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

- “Versioni IBM MQ disponibili” a pagina 35
- “Versioni Red Hat OpenShift Container Platform compatibili” a pagina 36
- “IBM Cloud Pak for Integration versioni” a pagina 36
- “Versioni IBM MQ disponibili in operatori meno recenti” a pagina 36
- “Versioni OpenShift Container Platform compatibili per gli operatori meno recenti” a pagina 36

Versioni IBM MQ disponibili

Canale operatore	Versione dell'operatore	IBM MQ versioni									
		9.4.3	9.4.2	9.4.1	9.4.0	9.3.5	9.3.4	9.3.3	9.3.2	9.3.1	9.3.0
v3.6	3.6	CD	DEP	DEP	MIG	DEP	DEP	DEP	DEP	DEP	MIG
v3.5	3.5		CD	DEP	MIG	DEP	DEP	DEP	DEP	DEP	MIG
v3.4	3.4			CD	MIG	DEP	DEP	DEP	DEP	DEP	MIG
v3.3	3.3			CD	MIG	DEP	DEP	DEP	DEP	DEP	MIG
v3.2-sc2	3.2				CD e SC2	DEP	DEP	DEP	DEP	DEP	MIG

Chiave:

- CD: il supporto Continuous Delivery è disponibile.
- SC2: IBM Cloud Pak for Integration - Support Cycle 2 (formerly Long Term Support) è disponibile.
- MIG: disponibile solo durante la migrazione da un operando IBM Cloud Pak for Integration - Support Cycle 2 (formerly Long Term Support) ad uno Continuous Delivery .
- DEP: Deprecated Obsoleto. Poiché le release IBM MQ non sono più supportate, potrebbero essere ancora configurabili nell'operatore, ma non sono più idonee per il supporto e potrebbero essere rimosse nelle release future.

Consultare “Cronologia delle release per IBM MQ Operator” a pagina 5 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	OpenShift Container Platform versioni[1]					
		4.18	4.17	4.16	4.15	4.14	4.12
v3.6	3.6.0 in poi	CD	CD	CD	CD	CD	CD
v3.5	3.5.1 in poi	CD	CD	CD	CD	CD	CD
	3.5.0 in avanti		CD	CD	CD	CD	CD
v3.4	3.3.4.0 in poi		CD	CD	CD	CD	CD
v3.3	3.3.3.0 in poi		CD	CD	CD	CD	CD
v3.2-sc2	3.2.10 in poi	SC2	SC2	SC2	SC2	SC2	SC2
	3.3.2.6 in poi		SC2	SC2	SC2	SC2	SC2
	3.2.3 in poi			SC2	SC2	SC2	SC2
	3.2.0 e versioni successive				SC2	SC2	SC2

Chiave:

- CD: il supporto Continuous Delivery è disponibile.
- SC2: IBM Cloud Pak for Integration - Support Cycle 2 (formerly Long Term Support) è disponibile.
- EOS: non più supportato. Eseguire la migrazione a una versione OpenShift Container Platform successiva.

IBM Cloud Pak for Integration versioni

Ciascuna versione di IBM MQ Operator è supportata per l'uso come parte di una versione di IBM Cloud Pak for Integration (come indicato nella tabella seguente) o indipendentemente.

IBM Cloud Pak for Integration Versione	IBM MQ Operator versioni
16.1.2	3.6.x
16.1.1	3.4.x, 3.5.x
16.1.0	3.2.x, 3.3.x

Versioni IBM MQ disponibili in operatori meno recenti

Vedi Available IBM MQ versions nella documentazione di IBM MQ 9.3 .

Versioni OpenShift Container Platform compatibili per gli operatori meno recenti

Consultare Compatible OpenShift Container Platform versions nella documentazione IBM MQ 9.3 .

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

OpenShift MQ Adv. Modifica delle risorse create da IBM MQ Operator

IBM MQ Operator riconcilia una risorsa personalizzata QueueManager creando e gestendo le risorse Kubernetes native. Queste risorse gestite **non devono** essere modificate direttamente.

Generalmente, è possibile determinare se una risorsa è di proprietà di un'altra risorsa di livello superiore, esaminando `ownerReferences`. Ad esempio, i seguenti metadati presi da un `StatefulSet` mostrano che appartengono alla risorsa `QueueManager` "qm1":

```
metadata:
  ownerReferences:
    - apiVersion: mq.ibm.com/v1beta1
      kind: QueueManager
      name: qm1
      uid: 60fda34c-9f7c-42d2-a293-78fec4315c62
      controller: true
      blockOwnerDeletion: true
```

Notare che non tutte le risorse hanno questi metadati.

È responsabilità di IBM MQ Operator gestire le risorse sottostanti, come `StatefulSet`, `Service` e `Route`. Se modifichi una di queste risorse sottostanti, IBM MQ Operator le modificherà e potresti riscontrare un tempo di inattività se tale modifica richiede un aggiornamento progressivo.

La maggior parte delle impostazioni importanti per i gestori code sono disponibili sulla risorsa `QueueManager` . Tuttavia, se si ritiene che sia necessario il controllo completo delle risorse sottostanti, sono disponibili alcune opzioni:

- Se devi sovrascrivere le impostazioni sul pod creato da IBM MQ Operator, puoi aggiungere un modello di sovrascrittura pod nella sezione `.spec.template` di `QueueManager` YAML.
- Se devi sovrascrivere le impostazioni sul gestore code `Route` creato da IBM MQ Operator, devi disabilitare l'impostazione dell'instradamento `.spec.route.enabled` su "false" e quindi creare il tuo percorso.
- Le impostazioni come le etichette e le annotazioni, così come le impostazioni Pod come `securityContext`, possono essere tutte impostate sulla risorsa `QueueManager` .
- In altri casi, il IBM MQ Operator potrebbe non essere appropriato per il tuo caso d'uso se hai bisogno di un controllo completo.

Pianificazione per la licenza di IBM MQ nei contenitori

La licenza del contenitore ti consente di concedere in licenza solo la capacità disponibile dei tuoi singoli contenitori IBM MQ , piuttosto che richiedere la licenza dell'intero server in cui sono in esecuzione i tuoi contenitori. Per sfruttare le licenze per container, è necessario utilizzare il sito IBM License Service per tenere traccia dell'utilizzo delle licenze e determinare l'abilitazione necessaria.

Si noti che la licenza basata su container è disponibile solo se si è su Red Hat OpenShift Container Platform o Kubernetes. Per ulteriori informazioni, consultare le FAQ sulle licenze per i container.

Per utilizzare il sito IBM License Service:

- Se si utilizza IBM MQ Operator, vedere il passo Distribuzione di License Service del processo di installazione di IBM MQ Operator .
- Se si sta distribuendo su Kubernetes, vedere Installazione di License Service sul cluster Kubernetes nella documentazione di IBM Cloud Pak foundational services .

Se non si utilizza Red Hat OpenShift Container Platform o Kubernetes, è necessario utilizzare IBM License Metric Tool. Questo strumento è necessario per misurare e segnalare l'utilizzo delle licenze dei prodotti IBM distribuiti in ambienti di virtualizzazione tradizionali per sfruttare le licenze di sottocapacità. Per ulteriori informazioni, consultare la documentazione di IBM License Metric Tool.

Riferimenti correlati

“IBM MQ annotazioni sulla licenza del contenitore” a pagina 281

Negli ambienti Kubernetes e Red Hat OpenShift Container Platform , IBM License Service usa le annotazioni di Pod per tracciare l'uso in base ai limiti definiti sul contenitore, piuttosto che sulla macchina sottostante. IBM MQ Operator imposta queste annotazioni di Pod per l'utente, ma quando si distribuisce con altri mezzi, è necessario impostarle da soli.

Informazioni correlate

Licenze di IBM Container

Domande frequenti sulle licenze dei contenitori

Installazione di License Service

Visualizzazione e traccia dell'utilizzo della licenza

OpenShift CP4I Kubernetes Pianificazione dell'archiviazione 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. Ciò comprende:

- 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 159). Se questo non viene completato, IBM MQ dovrà essere configurato ogni volta che il contenitore viene riavviato.

OpenShift CP4I Ad esempio, per configurare IBM MQ con memoria effimera, il tipo di archiviazione di `QueueManager` deve includere quanto segue:

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

Storage persistente

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. Questo è il funzionamento predefinito. Poiché ci sono diversi provider di memoria, ognuno dei quali supporta diverse funzionalità, ciò spesso significa che è richiesta la personalizzazione della configurazione. Il seguente esempio descrive i campi comuni che personalizzano la configurazione dell'archivio di IBM MQ nell'API v1beta1 :

- **`spec.queueManager.availability`** controlla la modalità di disponibilità. Se si utilizza `SingleInstance` o `NativeHA`, è necessaria solo l'archiviazione `ReadWriteOnce` . Per `multiInstance` è necessaria una classe di memoria che supporti `ReadWriteMany` con le caratteristiche di blocco file corrette. IBM MQ fornisce un' istruzione di supporto e una istruzione di test. La modalità di disponibilità influenza anche il layout del volume persistente. Per ulteriori informazioni, fare riferimento a “Pianificazione dell'alta disponibilità per IBM MQ nei contenitori” a pagina 40.
- **`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: [65534] # Change to 99 for clusters with RHEL7 or earlier worker nodes
```

Per informazioni sulle considerazioni sulla memoria per i gestori code della HA nativa, consultare “HA nativa” a pagina 42.

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

Capacità di memoria

Quando si utilizza IBM MQ Operator , è necessario provare ad assicurarsi di richiedere volumi sufficientemente grandi per le esigenze in corso. Tuttavia, se è necessario aumentare la capacità di memoria di uno o più volumi, questi volumi possono essere espansi se la classe di memoria supporta l'espansione dei volumi. I volumi possono essere espansi mediante una procedura online o offline. Una procedura offline richiede il riavvio dei pod `QueueManager` , mentre una procedura online non lo richiede. Per determinare se la classe di memoria supporta l'espansione del volume e quale procedura seguire per l'espansione del volume, fare riferimento alla documentazione del provider di memoria. È necessario considerare queste informazioni quando si seleziona una classe di memoria. Per una guida all'espansione del volume, consultare “Espansione dei volumi persistenti” a pagina 167.

Crittografia

IBM MQ non crittografa attivamente i dati inattivi. Pertanto, è necessario utilizzare l'archiviazione codificata passivamente o IBM MQ Advanced Message Security, o entrambi, per codificare i messaggi. Su IBM Cloud sia l'archiviazione blocchi che l'archiviazione file sono disponibili con la crittografia passiva inattiva.

Pianificazione dell'alta disponibilità per IBM MQ nei contenitori

Esistono tre opzioni per l'alta disponibilità con IBM MQ Operator: **gestore code HA nativo** (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 di rete) o **Gestore code resiliente singolo** (che offre un approccio semplice per l'HA utilizzando la memoria di 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

I gestori code della HA nativa coinvolgono un **attivo** e due pod di **replica** Kubernetes , che vengono eseguiti come parte di uno Kubernetes StatefulSet con esattamente tre repliche ciascuna con la propria serie di volumi persistenti Kubernetes . I requisiti IBM MQ per i file system condivisi si applicano anche quando si utilizza un gestore code HA nativo (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 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 42.

Replica nativa HA CRR (Cross-Region Replication)

Puoi implementare una soluzione di ripristino di emergenza che comprende due gruppi HA nativi ubicati in regioni separate. Un gruppo HA nativo ha il ruolo **live**, mentre l'altro gruppo ha il ruolo di **recupero**. Quando si verifica un'interruzione pianificata o non pianificata nella regione in cui si trova il gruppo live, l'azione dell'utente può far passare il ruolo del gruppo nella regione remota da recovery a live. Questo gruppo prende quindi il controllo del lavoro dell'istanza del gestore code originale, accettando le connessioni dell'applicazione e così via.

Per ulteriori informazioni, consultare “Replica nativa HA tra regioni diverse” a pagina 46.

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 questo, si utilizzano i volumi persistenti 'Kubernetes 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 usa un valore '**`imagePullPolicy`** di `IfNotPresent`, l'immagine potrebbe essere già disponibile su quel 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 stateful come 'IBM MQ con un 'Kubernetes 'StatefulSet,, se un nodo master 'Kubernetes perde il contatto con un nodo worker, non può determinare se il nodo è fallito 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.

Nota: La modifica dei dettagli dello StatefulSet di un gestore code IBM MQ , incluso il numero di repliche, non è supportata quando il gestore code viene creato tramite il IBM MQ Operator.

Concetti correlati

Configurazioni HA (High Availability)

Attività correlate

“Configurazione dell'alta disponibilità per i gestori code utilizzando IBM MQ Operator” a pagina 111
è possibile configurare l'alta disponibilità per i gestori di code utilizzando la soluzione Native HA o la soluzione per gestori di code multistanza.

CP4I MQ Adv. Kubernetes 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 di memoria 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.

Per garantire prestazioni e affidabilità, si consiglia l'uso di uno storage persistente RWOReadWriteOnce) con una configurazione Native HA. I volumi RWO da qualsiasi provider di memoria sono supportati se soddisfano le condizioni riportate di seguito:

- Ottenuto da un provider di archiviazione blocchi.
- Formattato come ext4 o XFS (che garantisce la conformità POSIX).
- Supporta il provisioning dinamico dei volumi e "volumeBindingMode: WaitForFirstConsumer".

I seguenti fornitori sono esplicitamente vietati:

- NFS
- GlusterFS
- Altri provider non di blocco.

Nota: V 9.4.2 Se si configura la replica cross-region di Native HA, sarà necessario almeno il doppio dello spazio su disco rispetto a Native HA. Ciò è dovuto al requisito di una directory di backup dei log durante il rebasing (vedere “Replica nativa HA tra regioni diverse” a pagina 46)

La seguente figura mostra una tipica distribuzione con tre istanze di un gestore code distribuite in tre contenitori.

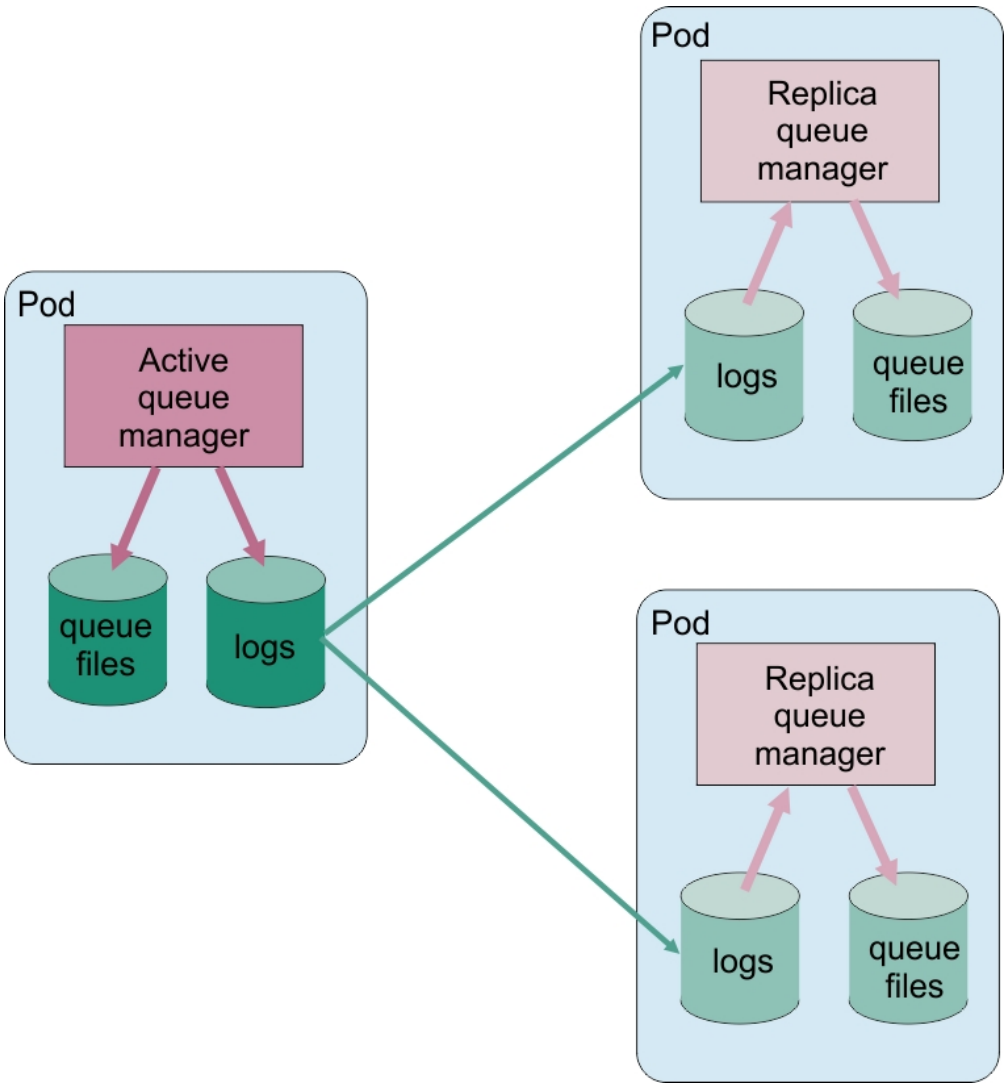


Figura 1. Esempio di configurazione della HA nativa

MQ Adv. 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.

Kubernetes 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 28
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.

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

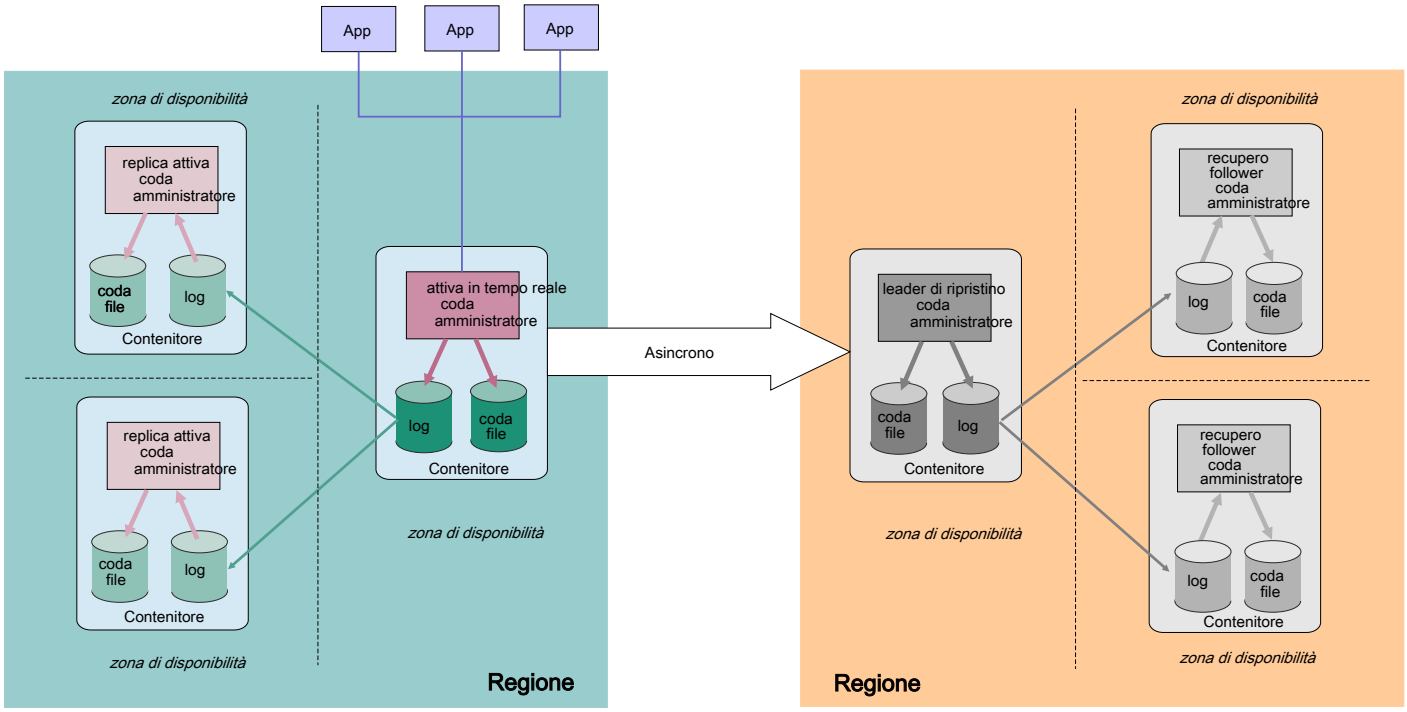
ha_for_ctr.dita

V 9.4.2 Replica nativa HA tra regioni diverse

È possibile implementare una configurazione di replica interregionale (CRR) Native HA basata sulla soluzione Native HA. La configurazione CRR fornisce il ripristino di emergenza.

Vedere “HA nativa” a pagina 42 per una panoramica della soluzione Native HA.

Native HA utilizza un gruppo di tre istanze di queue manager per mantenere una copia replicata dei dati di log. Il gruppo elegge automaticamente un'istanza attiva confrontando e scegliendo la migliore copia dei dati. Il CRR estende la replica dei log per includere un gruppo (regione) potenzialmente geograficamente distante che è anche altamente disponibile. Quando si verifica un'interruzione pianificata o non pianificata presso la sede principale, il gruppo presso la sede di recupero può subentrare nel lavoro. Inizialmente si definisce il ruolo di un gruppo, quindi si cambia manualmente il ruolo di un gruppo come richiesto per completare un passaggio pianificato o quando si verifica un'interruzione imprevista. I dati di registro vengono replicati al gruppo di ripristino in modo asincrono, pertanto potrebbe verificarsi una perdita di dati quando si verifica una commutazione a seguito di un'interruzione non pianificata.



Ruoli richiesti e ruoli effettivi

Un ruolo richiesto è l'impostazione di configurazione dell' GroupRole e per un'istanza Native HA specificata nella sezione " `NativeHALocalInstance` " del suo " `qm.ini`". L'impostazione richiede che un'istanza cooperi con le altre istanze del suo gruppo per svolgere un ruolo. Una volta che la maggioranza dei casi condivide lo stesso ruolo richiesto, può avvenire un'elezione per scegliere un leader del gruppo. Quando si vince un'elezione, il ruolo richiesto viene confrontato con l'ultimo ruolo utilizzato per ricavare un ruolo effettivo. Vedere la stanza NativeHAInstance del file qm.ini e la stanza NativeHALocalInstance del file qm.ini.

Ruolo dal vivo

Un gruppo Live è un normale gestore di code Native HA composto da tre istanze. Un'istanza viene scelta come attiva e questa accetta il nuovo lavoro dalle applicazioni. Un gestore di code Native HA esistente può essere migrato a una configurazione Native CRR.

Se un ruolo richiesto non è esplicitamente specificato nell' `qm.ini`, il valore predefinito è considerato Live. Un leader eletto di un gruppo Live è anche noto come gestore di coda attivo, questo istanza può accettare connessioni di applicazioni e strumenti per eseguire nuovi lavori di messaggistica e gestire oggetti di gestione coda.

Ruolo di recupero

Un gruppo di recupero è composto anche da tre istanze e una di queste viene eletta leader. Il leader agisce solo come delegato, accettando il lavoro inviato da un gruppo Live accoppiato. Non sono consentiti collegamenti di applicazioni al gruppo Recovery, sebbene siano consentiti alcuni comandi di controllo in modo che operazioni come la chiusura dell'istanza del gestore di coda possano essere completate.

La riga " `NativeHARecoveryGroup` " nel file di configurazione " `qm.ini` " del gruppo Live identifica l'indirizzo di rete per il gruppo Recovery.

In attesa di cambiamenti di ruolo

Quando si esegue una commutazione pianificata in cui l'attuale gruppo Live diventa il nuovo gruppo Recovery (e l'attuale gruppo Recovery diventa il nuovo gruppo Live) è necessario che ci sia coordinamento tra i gruppi. Questo coordinamento garantisce che i registri siano identici e raggiunga un RPO (recovery point objective) pari a zero.

Quando due gruppi coordinano il passaggio tra Recupero e Live, entrambi i gruppi assumono temporaneamente un ruolo di Pending efficace.

Ruolo di recupero a Ruolo live (in attesa di live)

Un gruppo che ha precedentemente operato in un ruolo di recupero e che intende passare a un ruolo live decide quando utilizzare il ruolo in base alla presenza o meno di un altro gruppo configurato.

Se non c'è nessun altro gruppo configurato, passa immediatamente al ruolo Live.

Se è configurato e abilitato un altro gruppo, il passaggio a un ruolo Live è bloccato finché l'altro gruppo non assume un ruolo Recovery o Pending Recovery e finché entrambi i gruppi non sono sincronizzati.

Ruolo live a ruolo di ripristino (ripristino in sospeso)

Un gruppo che in precedenza ha operato in un ruolo Live e che intende passare a un ruolo Recovery, deciderà quando utilizzare il ruolo Recovery in base alla presenza o meno di un altro gruppo configurato.

Se è configurato e abilitato un altro gruppo, il passaggio a un ruolo di ripristino è bloccato finché l'altro gruppo non assume un ruolo attivo o in attesa di attivazione e finché entrambi i gruppi non sono sincronizzati.

Replica e riorganizzazione

Mentre i due gruppi in una configurazione Native HA CRR sono connessi, gli aggiornamenti del registro vengono passati al leader del gruppo di ripristino e inoltrati alle altre istanze di ripristino. Questo processo è noto come **replica**.

Se la connessione di rete tra i due gruppi viene persa, potrebbe accadere che il gruppo di ripristino non riesca a recuperare utilizzando i registri replicati (ad esempio, perché il gruppo live ha riutilizzato le estensioni dei registri che sarebbero necessarie per il recupero). In questo caso, il responsabile del recupero esegue **un rebase**. Un rebase è il processo di scarto dei dati di log ricevuti e di ricostruzione del gestore della coda da un set completo di dati di log inviati dal gruppo live. Una volta che il leader del ripristino ha completato il rebase, i log vengono replicati nelle istanze di ripristino.

Se si verifica un problema durante un'operazione di rebase, è possibile che il gestore della coda non sia in grado di avviare nessuno dei membri del gruppo di ripristino. Per evitare questa possibilità, viene eseguito un backup del registro prima dell'inizio dell'operazione di rebase. Il gestore della coda può quindi essere riavviato utilizzando i dati del registro di backup se il rebase fallisce. Il registro di backup viene eliminato dopo che il rebase è riuscito.

A causa della necessità di un registro di backup, una configurazione Native HA CRR richiede almeno il doppio dello spazio di archiviazione di una configurazione Native HA.

OpenShift CP4I Pianificazione della sicurezza per IBM MQ nei contenitori

Considerazioni sulla sicurezza quando pianifichi il tuo IBM MQ nella configurazione dei contenitori.

Procedura

- “Autenticazione e autorizzazione utente per IBM MQ nei contenitori” a pagina 49
  - “Vincoli di sicurezza sull'utilizzo degli utenti del sistema operativo nei contenitori” a pagina 50
- “Considerazioni per limitare il traffico di rete a IBM MQ nei contenitori” a pagina 50

Autenticazione e autorizzazione utente per IBM MQ nei contenitori

IBM MQ nei contenitori può essere configurato per autenticare gli utenti tramite LDAP, Mutual TLS o un plugin MQ personalizzato.

Tieni presente che l'operatore IBM MQ non consente l'uso di utenti e gruppi del sistema operativo all'interno dell'immagine del contenitore. Per ulteriori informazioni, consultare “Vincoli di sicurezza sull'utilizzo degli utenti del sistema operativo nei contenitori” a pagina 50.

LDAP

Per informazioni sulla configurazione di IBM MQ per l'utilizzo di un repository utente LDAP, consultare Autenticazione connessione: Repository utente e Autorizzazione LDAP.

TLS reciproco

Se si configurano le connessioni in entrata a un gestore code per richiedere un certificato TLS (TLS reciproco), è possibile associare il DN del certificato a un nome utente. Devi fare due cose:

- Configurare un record di autenticazione di canale per creare l'associazione a un nome utente, utilizzando SSLPEER. Per ulteriori informazioni, consultare Associazione di un DN (Distinguished Name) SSL o TLS a un ID utente MCAUSER.
- Configurare il gestore code per definire i record di autorizzazioni per un nome utente non riconosciuto dal sistema. Per ulteriori informazioni, consultare Stanza di servizio del file qm.ini.

Token Web JSON

Per informazioni sulla configurazione di IBM MQ per utilizzare JWT (JSON Web Tokens), vedi Gestione dei token di autenticazione.

Plug-in MQ personalizzato

Questa è una tecnica avanzata, e richiede molto più lavoro. Per ulteriori informazioni, vedi Utilizzo di un servizio di autorizzazione personalizzato.

Attività correlate

"Esempio: configurazione di un gestore code con autenticazione TLS reciproca" a pagina 106
Questo esempio distribuisce un gestore code in OpenShift Container Platform utilizzando IBM MQ Operator. Il TLS reciproco viene utilizzato per l'autenticazione, per eseguire l'associazione da un certificato TLS a un'identità nel gestore code.

Vincoli di sicurezza sull'utilizzo degli utenti del sistema operativo nei contenitori

L'utilizzo di utenti del sistema operativo nei contenitori non è consigliato e non è consentito con l'operatore IBM MQ .

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 la password degli utenti - utilizza un programma "setuid" in modo da diventare l'utente "root" per eseguire questa operazione.

OpenShift CP4I Per garantire la conformità a queste misure di sicurezza, IBM MQ Operator non consente l'utilizzo di ID definiti nelle librerie del sistema operativo all'interno di un contenitore. Nessun ID utente o gruppo `mqm` definito nel contenitore.

Considerazioni per limitare il traffico di rete a IBM MQ nei contenitori

Puoi definire le politiche di rete per limitare il traffico ai pod nel tuo cluster in OpenShift Container Platform e Kubernetes. Questo argomento descrive alcune considerazioni su come le politiche di rete possono essere applicate a IBM MQ.

Per l'ingresso di rete in un gestore code, ci sono diverse porte da prendere in considerazione:

- Porta 1414 per il traffico del gestore code
- Porta 9414 per HA nativo
- Porta 9157 per le metriche
- Porta 9443 per la console web e API REST

L'uscita dalla rete è più complessa. Esempi di uscita di rete che si potrebbe voler considerare:

- DNS - se si dispone di canali o altre configurazioni che utilizzano nomi DNS
- Altri gestori code
- OCSP (Online Certificate Status Protocol) e CRL (Certificate Revocation Lists) - determinati dal provider di certificati.
- Provider di autenticazione:
  - LDAP
  - Aprire ID Connect o un altro provider di login configurato per il server Web IBM MQ . Ciò include IBM Cloud Pak Keycloak.

- Provider di traccia:
  - IBM Instana

  Nota: Per le versioni precedenti di IBM MQ , IBM Cloud Pak for Integration Operations Dashboard era disponibile anche come provider di traccia. Tuttavia, il dashboard Operazioni è stato rimosso in IBM MQ 9.3.3 CD e IBM MQ 9.4.0 LTS.

Esempio di NetworkPolicy

Di seguito è riportata una politica di rete di esempio per controllare l'ingresso per un gestore code denominato "myqm", da utilizzare su Red Hat OpenShift Container Platform.

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: myqm
spec:
  podSelector:
    matchLabels:
      app.kubernetes.io/instance: myqm
      app.kubernetes.io/name: ibm-mq
  ingress:
    # Allow access to queue manager listener from anywhere
    - ports:
        - protocol: TCP
          port: 1414
    # Allow access to Native HA port from other instances of the same queue manager
    - from:
        - podSelector:
            matchLabels:
              app.kubernetes.io/instance: myqm
              app.kubernetes.io/name: ibm-mq
      ports:
        - protocol: TCP
          port: 9414
    # Allow access to metrics from monitoring project
    - from:
        - namespaceSelector:
            matchLabels:
              network.openshift.io/policy-group: monitoring
      ports:
        - protocol: TCP
          port: 9157
    # Allow access to web server via Route
    - from:
        - namespaceSelector:
            matchLabels:
              network.openshift.io/policy-group: ingress
      ports:
        - protocol: TCP
          port: 9443
```

FIPS conformità per IBM MQ in contenitori

All'avvio, IBM MQ nei contenitori rileva se il sistema operativo su cui si sta avviando il contenitore è conforme a FIPS e (in caso affermativo) configura automaticamente il supporto a FIPS . I requisiti e le limitazioni sono indicati qui.

Federal Information Processing Standards

Il governo degli Stati Uniti fornisce consulenza tecnica sui sistemi IT e sulla sicurezza, inclusa la crittografia dei dati. Il National Institute for Standards and Technology (NIST) è un ente governativo che si occupa dei sistemi informatici e della sicurezza. Il NIST produce raccomandazioni e standard, tra cui il sito Federal Information Processing Standards (FIPS).

Uno standard significativo di FIPS è FIPS 140-2 , che richiede l'uso di algoritmi crittografici forti. FIPS 140-2 specifica anche i requisiti per gli algoritmi di hashing da utilizzare per proteggere i pacchetti da modifiche in transito.

IBM MQ fornisce il supporto a FIPS 140-2 se è stato configurato in tal senso.

Nota: Su AIX, Linux, and Windows, IBM MQ fornisce la conformità a FIPS 140-2 attraverso il modulo crittografico IBM Crypto for C (ICC) . Il certificato per questo modulo è stato spostato nello stato cronologico. I clienti devono prendere visione del certificato IBM Crypto for C (ICC) ed essere consapevoli di qualsiasi consiglio fornito dal NIST.

V 9.4.2 Il modulo crittografico FIPS 140-3 all'interno di IBM Semeru Runtime è stato approvato dal NIST nell'agosto 2024. IBM MQ 9.4.2 aggiunge il supporto per la gestione delle connessioni client di IBM MQ classes for JMS e IBM MQ classes for Java utilizzando TLS per FIPS 140-3 in Java 8 e IBM Semeru Runtime 11+. La certificazione NIST associata al modulo FIPS 140-3 può essere consultata all'indirizzo https://csrc.nist.gov/projects/cryptographic-module-validation-program/certificate/4755. Il provider FIPS 140-2 è ancora il profilo predefinito. IBM MQ 9.4.2 non modifica il comportamento predefinito, ma consente di configurare le connessioni con FIPS 140-3.

Per IBM MQ in Containers, l'immagine del contenitore IBM MQ Operator 3.2.0 e del gestore di code 9.4.0.0 in poi sono basati su UBI 9. FIPS 140-3 la conformità per IBM MQ in Containers è attualmente in attesa di essere soddisfatta.

Requisiti

Per i requisiti relativi alla configurazione dei cluster e altre considerazioni, vedere FIPS Wall: Approccio attuale IBM alla conformità FIPS .

IBM MQ nei container possono essere eseguiti in modalità di conformità a FIPS 140-2 . Durante l'avvio, IBM MQ nei container rileva se il sistema operativo host su cui si sta avviando il container è conforme a FIPS . Se il sistema operativo host è conforme a FIPS e sono stati forniti chiavi private e certificati, il contenitore IBM MQ configura il gestore di code, il server web IBM MQ e il trasferimento dei dati tra i nodi in un'implementazione Native High Availability, in modo da funzionare in modalità di conformità a FIPS .

Quando si utilizza IBM MQ Operator per distribuire i gestori code, l'operatore crea un instradamento con tipo di terminazione **Passthrough**. Ciò significa che il traffico viene inviato direttamente alla destinazione senza che il router fornisca la terminazione TLS. Il gestore di code IBM MQ e il server web IBM MQ sono le destinazioni in questo caso e forniscono già una comunicazione sicura conforme a FIPS .

Requisiti chiave:

1. Una chiave privata e i certificati, forniti in un segreto al gestore code e al server Web, che consentono ai client esterni di connettersi in modo sicuro al gestore code e al server web.
2. Una chiave privata e i certificati per trasferimento dati tra nodi differenti in una configurazione Native High Availability.

Limitazioni

Per una distribuzione conforme a FIPS di IBM MQ in container, considerare quanto segue:

- IBM MQ nei contenitori fornisce un endpoint per la raccolta di metriche. Attualmente questo endpoint è solo HTTP. È possibile disattivare l'endpoint delle metriche per rendere conforme il resto di IBM MQ FIPS .
- IBM MQ nei contenitori consente sovrascritture di immagini personalizzate. In altre parole, puoi creare immagini personalizzate utilizzando l'immagine del contenitore IBM MQ come immagine di base. FIPS la conformità potrebbe non essere applicabile a tali immagini personalizzate.
- Per il tracciamento dei messaggi tramite IBM Instana, la comunicazione tra IBM MQ e IBM Instana avviene tramite HTTP o HTTPS, senza la conformità con FIPS .
- IBM MQ Operator l'accesso ai servizi di gestione dell'identità e dell'accesso (IAM)/Zen di IBM non è conforme a FIPS .

Come viene rilevata la conformità a FIPS e come viene configurato automaticamente il supporto a FIPS

Se il sistema operativo su cui si avvia il contenitore è conforme a FIPS , il supporto di FIPS viene configurato automaticamente.

Nota: Su AIX, Linux, and Windows, IBM MQ fornisce la conformità a FIPS 140-2 attraverso il modulo crittografico IBM Crypto for C (ICC) . Il certificato per questo modulo è stato spostato nello stato cronologico. I clienti devono prendere visione del certificato IBM Crypto for C (ICC) ed essere consapevoli di qualsiasi consiglio fornito dal NIST.

V 9.4.2 Il modulo crittografico FIPS 140-3 all'interno di IBM Semeru Runtime è stato approvato dal NIST nell'agosto 2024. IBM MQ 9.4.2 aggiunge il supporto per la gestione delle connessioni client di IBM MQ classes for JMS e IBM MQ classes for Java utilizzando TLS per FIPS 140-3 in Java 8 e IBM Semeru Runtime 11+. La certificazione NIST associata al modulo FIPS 140-3 può essere consultata all'indirizzo https://csrc.nist.gov/projects/cryptographic-module-validation-program/certificate/4755. Il provider FIPS 140-2 è ancora il profilo predefinito. IBM MQ 9.4.2 non modifica il comportamento predefinito, ma consente di configurare le connessioni con FIPS 140-3.

Per IBM MQ in Containers, l'immagine del contenitore IBM MQ Operator 3.2.0 e del gestore di code 9.4.0.0 in poi sono basati su UBI 9. FIPS 140-3 la conformità per IBM MQ in Containers è attualmente in attesa di essere soddisfatta.

Durante l'avvio, IBM MQ nei container rileva se il sistema operativo su cui si sta avviando il container è conforme a FIPS . In tal caso, vengono eseguite automaticamente le seguenti operazioni:

Gestore code
Se il sistema operativo host è conforme a FIPS e la chiave privata e i certificati sono forniti, l'attributo di queue manager **SSLFIPS** è impostato su YES. Altrimenti, l'attributo **SSLFIPS** è impostato su NO.

IBM MQ Server Web
Il server web IBM MQ fornisce un'interfaccia HTTP / HTTPS per l'amministrazione di IBM MQ. Se il sistema operativo host è conforme a FIPS , le opzioni della JVM vengono aggiornate per far sì che il server Web utilizzi la crittografia conforme a FIPS . Per poter utilizzare FIPS, è necessario fornire la chiave privata e i certificati durante l'avvio del contenitore.

HA nativa
La sicurezza dei dati replicati tra nodi è controllata dalla sezione **NativeHALocalInstance** del file `qm.ini` . Ad esempio:

```
NativeHALocalInstance:
    KeyRepository=/run/runmqserver/ha/tls/key.kdb
    CertificateLabel=NHAQM
    CipherSpec=ECDHE_RSA_AES_256_GCM_SHA384
```

Se FIPS è abilitato, l'attributo **SSLFipsRequired** viene aggiunto alla stanza, con il valore impostato su `Sì` :

```
NativeHALocalInstance:
    KeyRepository=/run/runmqserver/ha/tls/key.kdb
    CertificateLabel=NHAQM
    CipherSpec=ECDHE_RSA_AES_256_GCM_SHA384
    SSLFipsRequired=Yes
```

Se il contenitore è in esecuzione in un cluster OpenShift senza supporto FIPS , i componenti Queue Manager, Web Server IBM MQ e Native HA non hanno il supporto FIPS automaticamente abilitato. Solo l'architettura x86-64 è attualmente supportata dalla piattaforma OpenShift per FIPS. Per le architetture Power e Linux on IBM Z , OpenShift non offre il supporto a FIPS . Per abilitare esplicitamente il supporto di FIPS nei componenti di IBM MQ per queste architetture, impostate la variabile d'ambiente *MQ_ENABLE_FIPS* su `true` nello YAML del gestore di code. Il seguente frammento YAML descrive l'uso della variabile d'ambiente *MQ_ENABLE_FIPS* :

```
template:
    pod:
      containers:
        - env:
            - name: MQ_ENABLE_FIPS
              value: "true"
          name: qmgr
```

Sovrascrittura della modalità automatica di FIPS per IBM MQ nei contenitori

Usare la variabile d'ambiente *MQ_ENABLE_FIPS* per abilitare o disabilitare esplicitamente la modalità FIPS per i componenti IBM MQ nel contenitore.

Prima di iniziare

Nota: Su AIX, Linux, and Windows, IBM MQ fornisce la conformità a FIPS 140-2 attraverso il modulo crittografico IBM Crypto for C (ICC) . Il certificato per questo modulo è stato spostato nello stato cronologico. I clienti devono prendere visione del certificato IBM Crypto for C (ICC) ed essere consapevoli di qualsiasi consiglio fornito dal NIST.

V 9.4.2 Il modulo crittografico FIPS 140-3 all'interno di IBM Semeru Runtime è stato approvato dal NIST nell'agosto 2024. IBM MQ 9.4.2 aggiunge il supporto per la gestione delle connessioni client di IBM MQ classes for JMS e IBM MQ classes for Java utilizzando TLS per FIPS 140-3 in Java 8 e IBM Semeru Runtime 11+. La certificazione NIST associata al modulo FIPS 140-3 può essere consultata all'indirizzo https://csrc.nist.gov/projects/cryptographic-module-validation-program/certificate/4755. Il provider FIPS 140-2 è ancora il profilo predefinito. IBM MQ 9.4.2 non modifica il comportamento predefinito, ma consente di configurare le connessioni con FIPS 140-3.

Per IBM MQ in Containers, l'immagine del contenitore IBM MQ Operator 3.2.0 e del gestore di code 9.4.0.0 in poi sono basati su UBI 9. FIPS 140-3 la conformità per IBM MQ in Containers è attualmente in attesa di essere soddisfatta.

Informazioni su questa attività

MQ_ENABLE_FIPS supporta tre valori:

automatico
Questo è il valore predefinito.

Se il sistema operativo host è abilitato a FIPS , tutti i componenti (queue manager, web server IBM MQ e Native HA) funzionano in modalità FIPS .

Se il sistema operativo host non è abilitato a FIPS , tutti i componenti non funzionano in modalità FIPS .

vero, true
Questo valore attiva FIPS per i componenti selezionati nel contenitore.

L'attributo del gestore di code **SSLFIPS** è impostato su YES anche se IBM MQ nei contenitori è in esecuzione su un sistema operativo host non conforme a FIPS . Cioè, se il gestore di code IBM MQ , il server web e l'HA nativo sono conformi a FIPS , ma il sistema operativo del contenitore non lo è.

No
Questo valore disattiva la conformità a FIPS .

L'attributo del gestore di code **SSLFIPS** è impostato su NO, anche se IBM MQ nei contenitori viene eseguito su una macchina host conforme a FIPS . Tuttavia, IBM MQ protegge comunque le connessioni se vengono forniti la chiave privata e i certificati.

Le opzioni JVM non vengono aggiornate per il server Web IBM MQ . Tuttavia, il server web IBM MQ esegue comunque un endpoint HTTPS se vengono forniti la chiave privata e i certificati.

La replica dei dati in Native HA non utilizza la crittografia FIPS .

Esempio

Ecco un esempio di YAML di queue manager che descrive l'abilitazione di TLS e FIPS per il componente queue manager.

Nota: Questo esempio utilizza una licenza IBM MQ Advanced , ma è possibile utilizzare anche una licenza IBM MQ . Per ulteriori informazioni, consultare “Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator” a pagina 160.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  namespace: ibm-mq-fips
  name: exampleqm
spec:
  license:
    accept: true
    license: L-NUUP-23NH8Y
    use: Production
  queueManager:
    name: EXAMPLEQM
    storage:
      queueManager:
        type: ephemeral
  template:
    pod:
      containers:
        - env:
            - name: MQ_ENABLE_FIPS
              value: "true"
          name: qmgr
  version: 9.4.3.0-r1
  web:
    enabled: false
  pki:
    keys:
      - name: ibm-mq-tls-certs
        secret:
          secretName: ibm-mq-tls-secret
          items:
          - tls.key
          - tls.crt
```

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. Da Red Hat OpenShift Container Platform 4.11, esiste un limite predefinito di 4096 processi per contenitore. Sebbene ciò sia adeguato per la maggior parte degli scenari, in alcuni casi potrebbe influire 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.

Preparazione, installazione e aggiornamento del tuo ambiente per IBM MQ nei contenitori

È possibile eseguire una gamma di attività per preparare il proprio ambiente per IBM MQ

Informazioni su questa attività

Se si utilizza IBM MQ Operator, preparare il cluster Red Hat OpenShift Container Platforminstallando l'operatore. Vedere “Installazione e aggiornamento di IBM MQ Operator” a pagina 56

Altrimenti, prepari il tuo ambiente contenitore creando le tue proprie immagini contenitore. Vedere “Preparazione per IBM MQ creando la propria immagine contenitore” a pagina 88

Installazione e aggiornamento di IBM MQ Operator

È possibile eseguire una serie di attività per installare, disinstallare e aggiornare IBM MQ Operator.

Informazioni su questa attività

Per iniziare a installare e aggiornare IBM MQ Operator su Red Hat OpenShift Container Platform, consultare i seguenti argomenti.

Procedura

OpenShift CP4I Dipendenze per IBM MQ Operator

Nessun altro operatore viene installato automaticamente quando si installa IBM MQ Operator.

IBM Licensing Operator deve essere installato separatamente per tenere traccia dell'utilizzo della licenza. Consultare Distribuzione del servizio di licenza License Service nella documentazione IBM Cloud Pak for Integration .

Quando si crea un `QueueManager` utilizzando una licenza IBM Cloud Pak for Integration , è possibile scegliere se si desidera o meno utilizzare SSO (single sign - on) con l'istanza IBM Cloud Pak for Integration

di Keycloak. L'utilizzo di Keycloak è abilitato per impostazione predefinita con una licenza IBM Cloud Pak for Integration , ma se non è installato, `QueueManager` entrerà in uno stato "Bloccato" fino a quando non saranno installate le dipendenze corrette. Consultare “Installazione del IBM MQ Operator” a pagina 58 per ulteriori dettagli sulle dipendenze.

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

Il sito IBM MQ Operator richiede sempre i permessi di accesso al cluster per leggere le informazioni sulla classe di archiviazione e sulla versione del cluster.

Il sito IBM MQ Operator può essere installato in due ambiti diversi:

1. L **'ambito del cluster** significa che all'operatore verranno assegnati i permessi per l'utilizzo delle risorse `ClusterRole` e `ClusterRoleBinding` . Questi consentono all'operatore di agire in tutti gli spazi dei nomi.
2. L' **ambito dello spazio dei nomi** significa che all'operatore verranno assegnati i permessi usando `Role` e `RoleBinding`, che daranno solo i permessi nello spazio dei nomi dato. Inoltre, ci saranno `ClusterRole` e `ClusterRoleBinding` per i seguenti permessi di sola lettura:

- 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. In questo modo l'operatore può tentare di segnalare eventuali problemi con le classi di stoccaggio selezionate nei contenitori.
  - Gruppi API: **`storage.k8s.io`**
  - Risorse: **`storageclasses`**
  - verbs: **`get, list`**

OpenShift CP4I Verifica delle firme di immagine

Le immagini del contenitore del gestore code IBM MQ Operator e IBM MQ sono firmate digitalmente.

Informazioni su questa attività

Le firme digitali forniscono un modo ai consumatori di contenuti per garantire che ciò che scaricano sia autentico (proviene dalla fonte prevista) e abbia integrità (è ciò che ci aspettiamo che sia).

Procedura

- Verificare le firme delle immagini del contenitore del gestore code IBM MQ Operator e IBM MQ :
  - Per un'immagine del contenitore del gestore di code 'IBM MQ Operator alla versione 3.4.0 o successiva, o per un'immagine del contenitore del gestore di code 'IBM MQ alla versione 9.4.1.0-r2 o successiva, vedere Verifica delle firme dell'immagine nella documentazione di 'IBM Cloud Pak for Integration (CP4I) 16.1.1.
  - Per un'immagine del contenitore del gestore di code 'IBM MQ Operator alla versione 3.2.x o 3.3.x, o per un'immagine del contenitore del gestore di code 'IBM MQ tra le versioni 9.4.0.0-r1 e 9.4.1.0-r1, vedere Verifica delle firme delle immagini nella documentazione di 'IBM Cloud Pak for Integration (CP4I) 16.1.0.

Installazione del IBM MQ Operator

Il IBM MQ Operator può essere installato su Red Hat OpenShift utilizzando la console OpenShift o la CLI (command line interface).

Prima di iniziare

Importante:

- Questo argomento è per l'installazione di IBM MQ Operator per uso autonomo **solo**. Se si intende utilizzare l'SSO IBM Cloud Pak for Integrationo Keycloak per uno o più gestori code, fare riferimento a “Installazione di IBM MQ Operator per l'utilizzo con CP4I” a pagina 65.
- Esamina le istruzioni su come strutturare la tua distribuzione prima di installare IBM MQ Operator.

Per assicurarsi che l'installazione avvenga nel modo più semplice possibile, accertarsi di aver compreso tutti i prerequisiti e i requisiti prima di avviare l'installazione. Consultare “Pianificazione per IBM MQ nei contenitori” a pagina 27.

Informazioni su questa attività

I seguenti passi rappresentano il tipico flusso di attività per installare IBM MQ Operator:

1. Installare Red Hat OpenShift Container Platform.
2. Configurare l'archiviazione.
3. Immagini mirror (solo air - gap).
4. Aggiungi il catalogo IBM MQ Operator.
5. Installa IBM MQ Operator.
6. Creare il segreto della chiave di titolarità (solo installazioni online).
7. Distribuire il servizio di licenza License Service.
8. Distribuire un gestore code.

Procedura

1. Installa Red Hat OpenShift Container Platform.

   Per la procedura dettagliata per l'installazione di OpenShift, consultare Installazione del software Red Hat 4.6 o successiva.

   Importante: Assicurarsi di installare una versione supportata di OpenShift Container Platform. Ad esempio, per utilizzare IBM MQ Operator 3.6 o versioni successive, è necessario installare OpenShift Container Platform 4.12 o versioni successive. Per ulteriori informazioni, vedere IBM Cloud Pak e Red Hat OpenShift Container Platform compatibilità.

   Per qualsiasi passo che utilizza la CLI Red Hat OpenShift Container Platform , devi essere collegato al tuo cluster OpenShift con `oc login`. Per installare la CLI, vedi Introduzione alla CLI OpenShift.

   Dopo aver installato OpenShift, puoi verificare e ottenere l'accesso al tuo software del contenitore utilizzando la IBM chiave di titolarità che crei in Crea il segreto della chiave di titolarità.
2. Configurare la memoria.

   È necessario definire le classi di memoria in Red Hat OpenShift Container Platform e impostare la propria configurazione di memoria per soddisfare i propri requisiti di dimensione.

   Importante: I gestori code IBM MQ a istanza singola e HA nativa possono utilizzare la modalità di accesso RWO, mentre i gestori code a più istanze richiedono RWX come descritto in “Pianificazione dell'archiviazione per IBM MQ Operator” a pagina 38. I gestori code a più istanze IBM MQ richiedono particolari caratteristiche del file system, che possono essere verificate utilizzando le istruzioni per Verifica di un file system condiviso per IBM MQ.

Un elenco di file system noti, conformi e non conformi, e note su altri limiti o limitazioni, è disponibile nell' istruzione di test per i file system IBM MQ.

I provider di memoria consigliati possono essere trovati nella pagina CP4I Considerazioni sulla memoria .

3. Immagini speculari (solo air - gap).

   Se il cluster si trova in un ambiente di rete con restrizioni (air-gapped), è necessario eseguire il mirroring delle immagini dell' IBM MQ .

   Importante: Dalla versione 3.5.0 l' IBM MQ CASE è cambiato e ora contiene una sola versione di queue manager. Il processo di mirroring delle immagini è notevolmente più veloce, ma potrebbe essere necessario eseguirlo più volte, una per ogni versione del gestore di code che si sta installando.

   Per ulteriori informazioni, consultare “IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

   Devi cercare quali versioni di IBM MQ CASE contengono le versioni di operatore e gestore di coda che ti servono e ripetere il processo di mirroring air-gap per ciascuna di esse. Vedere l' IBM MQ CASE Lookup Tables for the version mappings.

   Ad esempio, quando si seguono le istruzioni collegate di seguito per eseguire il mirroring delle immagini, si installa l'ultima versione dell'operatore e della singola versione corrispondente del gestore della coda utilizzando i seguenti valori:

   ```
   export OPERATOR_PACKAGE_NAME=ibm-mq
   export OPERATOR_VERSION=3.6.0
   ```

   Per informazioni sul contenuto di IBM MQ CASE, utilizzato per le installazioni e gli aggiornamenti air-gap, vedere '“IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

   Per creare immagini di mirroring, consultare Mirroring delle immagini per un cluster air - gapped.

4. Aggiungere l'origine del catalogo IBM MQ Operator .

   Aggiungi l'origine del catalogo che rende gli operatori disponibili al tuo cluster. Consultare “Aggiunta dell'origine del catalogo IBM MQ Operator” a pagina 60.

5. Installare IBM MQ Operator.

   Scegli una delle seguenti due opzioni (utilizza la console o utilizza la CLI):

   - Opzione 1: Installare IBM MQ Operator utilizzando la console OpenShift.
   - Opzione 2 Installa IBM MQ Operator utilizzando la CLI OpenShift.

6. Creare il segreto della chiave di titolarità (solo installazioni in linea).

   IBM MQ Operator distribuisce le immagini del gestore code estratte da un registro del contenitore che esegue un controllo di titolarità della licenza. Questo controllo richiede una chiave di titolarità memorizzata in un segreto di pull `docker-registry` . Se non si dispone ancora di una chiave di titolarità nello spazio dei nomi in cui verranno installati i gestori code, seguire queste istruzioni per ottenere una chiave di titolarità e creare un segreto di pull.

   Nota: La chiave di titolarità non è richiesta se verranno distribuiti solo i gestori code IBM MQ Advanced for Developers (non Warranted).

   Puoi creare il segreto della chiave di titolarità utilizzando la console OpenShift o la CLI. Il seguente esempio utilizza la CLI:

   a. Ottieni la chiave di titolarità assegnata al tuo ID IBM . 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.

c. Dalla CLI OpenShift , immetti il seguente comando per creare un segreto di pull dell'immagine denominato `ibm-entitlement-key`.

```
oc create secret docker-registry ibm-entitlement-key \
--docker-server=cp.icr.io \
--docker-username=cp \
--docker-password=entitlement_key \
--docker-email=user_email
\--namespace=namespace
```

Dove *entitlement_key* è la chiave di titolarità che hai copiato nel passaggio b, *user_email* è l' IBM ID associato al software autorizzato e *namespace* è lo spazio dei nomi in cui hai installato il tuo IBM MQ Operator .

7. Distribuire License Service.

   Ciò è necessario per il monitoraggio dell'utilizzo della licenza dei gestori code. Seguire le istruzioni riportate in Distribuzione del servizio di licenza License Service.

8. Distribuire un gestore code.

   Per istruzioni sulla distribuzione di un gestore code di esempio "avvio rapido" , consultare “Distribuzione di un gestore code semplice utilizzando IBM MQ Operator” a pagina 98.

Attività correlate

“Disinstallazione di IBM MQ Operator” a pagina 84
Puoi utilizzare la console o la CLI Red Hat OpenShift per disinstallare IBM MQ Operator da Red Hat OpenShift.

OpenShift *Aggiunta dell'origine del catalogo IBM MQ Operator*

Aggiungere l'origine del catalogo IBM MQ Operator al proprio cluster OpenShift per rendere IBM MQ Operator disponibile per l'installazione. Questa attività è richiesta anche se si applicano i fix pack di origine del catalogo prima di completare un aggiornamento.

Informazioni su questa attività

Un catalogo di operatori è un indice di operatori disponibili per estendere l'API di un cluster Red Hat OpenShift Container Platform per abilitare i prodotti software IBM .

Sono disponibili le seguenti origini catalogo:

Opzione 1: origine catalogo specifica per IBM MQ Operator.

Utilizzando un'origine del catalogo IBM MQ Operator specifica, si ottiene il controllo completo della versione del software su un cluster e quando si verificano gli aggiornamenti. Una nuova versione di IBM MQ Operator diventa disponibile **solo** in un cluster OpenShift dopo aver aggiornato l'origine del catalogo. Questo processo fornisce in modo efficace il controllo manuale degli aggiornamenti, quindi non è necessario utilizzare l'opzione `Manuale` per l'impostazione **`Update approval`** per gli operatori. L'opzione **Manuale** forza l'esecuzione di tutti i possibili aggiornamenti contemporaneamente e può bloccare gli aggiornamenti, quindi utilizzare solo l'opzione **Automatico** . Per ulteriori informazioni, consulta la sezione "Limitazione degli aggiornamenti automatici con una strategia di approvazione" di Installazione degli operatori utilizzando la console Red Hat OpenShift.

Scegliere questa opzione se si sta completando un aggiornamento e si deve aggiungere l'origine del catalogo IBM MQ Operator di una versione più recente.

Per utilizzare questa opzione, passare a Opzione 1: aggiungere origini di catalogo specifiche per IBM MQ Operator.

Opzione 2: IBM Operator Catalog.

Con questa opzione, le nuove versioni dell'operatore diventano disponibili e vengono applicate **senza** alcun intervento da parte dell'utente. Utilizzare questa opzione **solo** per le installazioni in linea in cui si desiderano gli aggiornamenti **automatici** di IBM MQ Operatore in cui le installazioni deterministiche non sono necessarie.

Nota: Questa opzione può essere utile per ambienti di prova, ma **non è adatta per ambienti di produzione**.

Per utilizzare questa opzione, passare a Opzione 2: aggiungere il IBM Catalogo operatore.

Procedura

- **Opzione 1: aggiungere origini di catalogo specifiche per IBM MQ Operator.**

  Questa attività presuppone che siano stati completati i primi 3 passi di “Installazione del IBM MQ Operator” a pagina 58.

  Questa attività deve essere eseguita da un amministratore cluster e deve essere eseguita utilizzando la CLI.

  a) Solo aggiornamento: se si stanno applicando i fix pack di origine del catalogo prima di un aggiornamento, completare la seguente procedura:
     - Confermare che gli operatori siano in esecuzione correttamente.
     - Se sono presenti aggiornamenti IBM MQ Operator in sospeso che richiedono l'approvazione manuale, approvarli prima di avviare questa procedura. Per ulteriori informazioni, vedi "Limitazione degli aggiornamenti automatici con una strategia di approvazione" in Installazione degli operatori utilizzando la console Red Hat OpenShift.

  b) Se non l'hai già installato o se è necessario aggiornarlo, scarica il plug-in IBM Catalog Management (versione 1.6.0 o successiva) da GitHub.

     Questo plug-in consente di eseguire i comandi **oc ibm-pak** sul cluster.

  c) Accedi al tuo cluster utilizzando il comando **oc login** e le credenziali utente:

     ```
     oc login openshift_url -u username -p password -n namespace
     ```

  d) Esportare le seguenti variabili di ambiente per IBM MQ Operator:

     ```
     export OPERATOR_PACKAGE_NAME=ibm-mq
     export OPERATOR_VERSION=CHOSEN_OPERATOR_VERSION
     export ARCH=ARCHITECTURE
     ```

     Dove
     - *ARCHITECTURE* è l'architettura del sistema su cui si sta distribuendo il file IBM MQ Operator e ha un valore di `amd64`, `ppc64le` o `s390x`.
     - *CHOSEN_OPERATOR_VERSION* è la versione richiesta di IBM MQ Operator. La versione attuale CD è 3.6.0. La versione attuale SC2 è 3.2.13.

     Importante: Se si sta spostando dal catalogo operatore IBM all'origine del catalogo specifico per IBM MQ Operator, impostare `OPERATOR_VERSION` sulla versione della distribuzione di IBM MQ Operator.

  e) Scaricare i file per l'operatore IBM MQ .

     Nota: Se si sta completando un'installazione **air - gapped** , è necessario disporre già dei file necessari dopo aver completato il passo "Immagini di mirroring" di "Installazione di IBM MQ Operator", nel qual caso è possibile passare al passo “8” a pagina 62 "Applicare l'origine del catalogo IBM MQ Operator al cluster".

     ```
     oc ibm-pak get ${OPERATOR_PACKAGE_NAME} --version ${OPERATOR_VERSION}
     ```

  f) Generare l'origine del catalogo richiesta per IBM MQ Operator:

     ```
     oc ibm-pak generate mirror-manifests ${OPERATOR_PACKAGE_NAME} icr.io --version $
     {OPERATOR_VERSION}
     ```

  g) Opzionale: Generare le origini del catalogo e salvarle in un'altra directory.

     a. Ottieni l'origine del catalogo:

```
cat ~/.ibm-pak/data/mirror/${OPERATOR_PACKAGE_NAME}/${OPERATOR_VERSION}/catalog-
sources.yaml
```

b. (Facoltativo) Passare alla directory nel browser di file per copiare queste risorse utente in file che è possibile conservare per il riutilizzo o per le pipeline.

h) Applica l'origine del catalogo IBM MQ Operator al cluster.

```
oc apply -f ~/.ibm-pak/data/mirror/${OPERATOR_PACKAGE_NAME}/${OPERATOR_VERSION}/catalog-
sources.yaml
```

i) Conferma che l'origine del catalogo IBM MQ Operator è stata creata nel namespace `openshift-marketplace` :

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

Output di esempio:

```
oc get catalogsource -n openshift-marketplace

NAME                           DISPLAY                TYPE   PUBLISHER   AGE
ibmmq-operator-catalogsource   ibm-mq-3.1.3           grpc   IBM         23h
```

Si è ora pronti a completare il Passo 5 dell'installazione di IBM MQ Operator.

- **Opzione 2: aggiungere il IBM Catalogo operatore.**

  Importante: Utilizzare il IBM Catalogo operatore **solo** per le installazioni in linea in cui si desiderano gli aggiornamenti **automatici** di IBM MQ Operatore in cui le installazioni deterministiche non sono necessarie. Questa opzione può essere utile per ambienti di prova, ma **non è adatta per ambienti di produzione**.

  Il IBM Operator Catalog è un indice di operatori disponibili per estendere l'API di un cluster Red Hat OpenShift Container Platform per abilitare i prodotti software IBM . L'aggiunta delle origini del catalogo al tuo cluster OpenShift aggiunge gli operatori IBM all'elenco di operatori che puoi installare.

  Questa attività presuppone che siano stati completati i primi 3 passi di “Installazione del IBM MQ Operator” a pagina 58.

  Questa attività può essere eseguita utilizzando la CLI o la console web OpenShift .

  Utilizzo della CLI

  1. Copiare la seguente definizione risorsa per gli operatori IBM in un file locale sul computer:

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

  2. Immetti il seguente comando. Sostituisci *filename*.yaml con il nome del file che hai creato nel passaggio precedente:

```
oc apply -f filename.yaml
```

  Utilizzo della console web OpenShift

  1. Accedi alla console web OpenShift con le tue credenziali di amministratore del cluster OpenShift .
  2. Nel banner, fai clic sul segno più ("+") per aprire la casella di dialogo **Importa YAML** .

Nota: Non è necessario selezionare un valore per **Progetto**. Il codice YAML nel passo successivo include già il valore corretto per `metadata:namespace`, che garantisce che l'origine del catalogo sia installata nel progetto corretto (spazio dei nomi).

3. Incollare la seguente definizione della risorsa nella finestra:

```
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
```

4. Fai clic su **Crea**.

Si è ora pronti a completare il Passo 5 dell'installazione di IBM MQ Operator.

OpenShift CP4I *Installazione di IBM MQ Operator utilizzando la console OpenShift*

IBM MQ Operator può essere installato su Red Hat OpenShift utilizzando OperatorHub.

Prima di iniziare

Questa attività presuppone che siano stati completati i passi da 1 a 4 di “Installazione del IBM MQ Operator” a pagina 58.

Procedura

1. Accedi alla tua console del cluster Red Hat OpenShift .
2. Dal riquadro di navigazione, fare clic su **Operatori** > **OperatorHub**.
   Viene visualizzata la pagina OperatorHub .
3. Nel campo **Tutti gli elementi** , immettere "IBM MQ".
   Viene visualizzata la voce di catalogo IBM MQ .
4. Selezionare **IBM MQ**.
   Viene visualizzata la finestra IBM MQ .
5. Fai clic su **Installa**.
   Viene visualizzata la pagina Operatore di installazione.
6. Immetti i seguenti valori:
   a) Impostare **Canale** sulla versione scelta.

      Consultare “Supporto versione per IBM MQ Operator” a pagina 35 per stabilire quale canale operatore scegliere.

   b) Impostare la **Modalità di installazione** su "uno spazio dei nomi specifico sul cluster" (che è possibile creare nel passo successivo) 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.

   c) Opzionale: Se si sceglie "uno spazio dei nomi specifico sul cluster", impostare lo **Spazio dei nomi** sul valore del progetto (spazio dei nomi) in cui si desidera installare l'operatore.

      Nota: Quando si utilizza la console per installare l'operatore, è possibile utilizzare uno spazio dei nomi esistente, lo spazio dei nomi predefinito fornito dall'operatore o creare un nuovo spazio dei nomi. Se si desidera creare un nuovo spazio dei nomi è possibile crearlo da questo modulo,

come segue: dal riquadro di navigazione, fare clic su **Home** > **Progetti**, selezionare **Crea progetto**, specificare il **Nome** del progetto (lo spazio dei nomi) che si desidera creare, quindi fare clic su **Crea**.

d) Impostare **Strategia di approvazione** su Automatico.

7. Fare clic su **Installa** e attendere l'installazione dell'operatore.

   Viene fornita una conferma al termine dell'installazione.

   Per verificare l'installazione, passare a **Operatori** > **Operatori installati**e selezionare il proprio progetto dall'elenco a discesa **Progetti** . Lo stato dell'operatore cambia in Riuscito quando l'installazione è completa.

Operazioni successive

Sei ora pronto a Creare il segreto della chiave di titolarità (passo 6 di “Installazione del IBM MQ Operator” a pagina 58).

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 la CLI (command line interface).

Prima di iniziare

Questa attività presuppone che siano stati completati i passi da 1 a 4 di “Installazione del IBM MQ Operator” a pagina 58.

Procedura

1. Accedi alla CLI (command line interface) Red Hat OpenShift utilizzando **`oc login`**.
2. Opzionale: Crea 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 operazione è necessaria solo se si desidera eseguire l'installazione in un determinato spazio dei nomi che non esiste già.

   Per creare un nuovo spazio dei nomi nella CLI, esegui questo comando:

   ```
   oc create namespace namespace_name
   ```

   Dove *namespace_name* è il nome dello spazio dei nomi che si desidera creare.
3. Visualizzare l'elenco degli operatori disponibili per il cluster da OperatorHub:

   ```
   oc get packagemanifests -n openshift-marketplace
   ```
4. Esaminare IBM MQ Operator per verificarne il **`InstallModes`** supportato e disponibile **`Channels`**.

   ```
   oc describe packagemanifests ibm-mq -n openshift-marketplace
   ```
5. Opzionale: Creare un **`OperatorGroup`**.

   Un **`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 corrisponda alla modalità **`InstallMode`**dell'operatore, `AllNamespaces` o `SingleNamespace` .

   Se l'operatore che si desidera installare utilizza la modalità `AllNamespaces` , lo spazio dei nomi `openshift-operators` dispone già di un **`OperatorGroup`** appropriato ed è possibile ignorare questo passo.

   Se l'operatore utilizza la modalità `SingleNamespace` e non si dispone già di un **`OperatorGroup`** appropriato, crearne uno immettendo il seguente comando:

```
cat << EOF | oc apply -f -
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: operatorgroup_name
  namespace: namespace_name
spec:
  targetNamespaces:
  - namespace_name
EOF
```

6. Consultare “Supporto versione per IBM MQ Operator” a pagina 35 per stabilire quale canale operatore scegliere.
7. Installare l'operatore.

   Utilizzare il seguente comando, modificando i seguenti valori:

   - Modificare *ibm_mq_operator_channel* in modo che corrisponda al canale della versione di IBM MQ Operator che si vuole installare.
   - Cambiare *nome_spazio_nomi* in **openshift-operators** se si usa la modalità "AllNamespaces", o nello spazio dei nomi in cui si vuole distribuire il IBM MQ Operator se si usa la modalità "SingleNamespace".
   - Cambiare *catalog_source_name* con il nome dell'origine del catalogo usato per l'installazione IBM MQ Operator, come aggiunto nella fase di installazione precedente “Aggiunta dell'origine del catalogo IBM MQ Operator” a pagina 60.

```
cat << EOF | oc apply -f -
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: ibm-mq
  namespace: namespace_name
spec:
  channel: ibm_mq_operator_channel
  installPlanApproval: Automatic
  name: ibm-mq
  source: catalog_source_name
  sourceNamespace: openshift-marketplace
EOF
```

8. Dopo pochi minuti, l'operatore viene installato. Eseguire il seguente comando per verificare che tutti i componenti siano nello stato Riuscito:

```
oc get csv -n namespace_name | grep ibm-mq
```

   Dove *nome_spazio_nomi* è **openshift-operators** se si usa la modalità "AllNamespaces", oppure il nome del progetto (spazio dei nomi) se si usa la modalità "SingleNamespace".

Operazioni successive

Sei ora pronto a Creare il segreto della chiave di titolarità (passo 6 di “Installazione del IBM MQ Operator” a pagina 58).

OpenShift CP4I Installazione di IBM MQ Operator per l'utilizzo con CP4I

Per l'utilizzo con IBM Cloud Pak for Integration (CP4I), IBM MQ Operator può essere installato su Red Hat OpenShift tramite la console OpenShift o la CLI (command line interface).

Prima di iniziare

Importante:

- Questo argomento è per l'installazione di IBM MQ Operator da utilizzare con CP4Io se si intende distribuire almeno uno dei gestori code utilizzando una CP4I licenza **solo**. Per istruzioni sull'installazione di IBM MQ Operator per uso autonomo, consultare “Installazione del IBM MQ Operator” a pagina 58.
- Esamina le istruzioni su strutturare la tua distribuzione prima di installare IBM MQ Operator.

Per assicurarsi che l'installazione avvenga nel modo più fluido possibile, accertarsi di aver compreso tutti i prerequisiti e i requisiti prima di avviare l'installazione. Consultare “Pianificazione per IBM MQ nei contenitori” a pagina 27.

Informazioni su questa attività

I seguenti passi rappresentano il tipico flusso di attività per l'installazione di IBM MQ Operator:

1. Installare Red Hat OpenShift Container Platform.
2. Configurare l'archiviazione.
3. Immagini mirror (solo air - gap).
4. Aggiungi il catalogo IBM MQ Operator e preparare il cluster.
5. Installa IBM MQ Operator.
6. Creare il segreto della chiave di titolarità (solo installazioni online).
7. Facoltativo: installare IBM Cloud Pak for Integration (CP4I) e le relative dipendenze.
8. Distribuire il servizio di licenza License Service.
9. Distribuire un gestore code.

Procedura

1. Installa Red Hat OpenShift Container Platform.

   Per la procedura dettagliata per l'installazione di OpenShift, consultare Installazione del software Red Hat 4.6 o successiva.

   Importante: Assicurarsi di installare una versione supportata di OpenShift Container Platform. Ad esempio, per utilizzare IBM MQ Operator 3.6 o versioni successive, è necessario installare OpenShift Container Platform 4.12 o versioni successive. Per ulteriori informazioni, consultare IBM Cloud Pak e la compatibilità Red Hat OpenShift Container Platform.

   Per qualsiasi passo che utilizza la CLI Red Hat OpenShift Container Platform , devi essere collegato al tuo cluster OpenShift con `oc login`. Per installare la CLI, vedi Introduzione alla CLI OpenShift.

   Dopo aver installato OpenShift, puoi verificare e ottenere l'accesso al tuo software del contenitore utilizzando la IBM chiave di titolarità che crei in Crea il segreto della chiave di titolarità.
2. Configurare la memoria.

   È necessario definire le classi di memoria in Red Hat OpenShift Container Platform e impostare la propria configurazione di memoria per soddisfare i propri requisiti di dimensione.

   Importante: I gestori code IBM MQ a istanza singola e HA nativa possono utilizzare la modalità di accesso RWO, mentre i gestori code a più istanze richiedono RWX come descritto in “Pianificazione dell'archiviazione per IBM MQ Operator” a pagina 38. I gestori code a più istanze IBM MQ richiedono particolari caratteristiche del file system, che possono essere verificate utilizzando le istruzioni per Verifica di un file system condiviso per IBM MQ.

   Un elenco di file system conformi e non conformi noti e note su altri limiti o limitazioni, è disponibile nell' istruzione di test per i file system IBM MQ.

   I provider di memoria consigliati possono essere trovati nella pagina CP4I Considerazioni sulla memoria .
3. Immagini speculari (solo air - gap).

   Se il cluster si trova in un ambiente di rete limitato (air-gapped), è necessario eseguire il mirroring delle immagini IBM MQ . A seconda della propria configurazione, potrebbe essere necessario anche eseguire il mirroring di alcuni componenti aggiuntivi. Leggere le seguenti informazioni, quindi eseguire il mirroring delle immagini come richiesto.

   - È necessario eseguire il mirroring delle immagini IBM MQ .

Importante: Dalla versione 3.5.0 l' IBM MQ CASE è cambiato e ora contiene una sola versione di queue manager. Il processo di mirroring delle immagini è notevolmente più veloce, ma potrebbe essere necessario eseguirlo più volte, una per ogni versione del gestore di code che si sta installando.

Per ulteriori informazioni, consultare “IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

Devi cercare quali versioni di IBM MQ CASE contengono le versioni di operatore e gestore di coda che ti servono e ripetere il processo di mirroring air-gap per ciascuna di esse. Vedere l' IBM MQ CASE Lookup Tables for the version mappings.

Ad esempio, quando si seguono le istruzioni collegate di seguito per eseguire il mirroring delle immagini, si installa l'ultima versione dell'operatore e la singola versione corrispondente del gestore della coda utilizzando i seguenti valori:

```
export OPERATOR_PACKAGE_NAME=ibm-mq
export OPERATOR_VERSION=3.6.0
```

Per informazioni sul contenuto di IBM MQ CASE, utilizzato per le installazioni e gli aggiornamenti air-gap, vedere '“IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

- È inoltre necessario eseguire il mirroring di alcuni componenti aggiuntivi richiesti se si intende distribuire almeno un gestore code in cui **tutte** le seguenti istruzioni sono vere:
  - Si sta utilizzando una licenza CP4I .
  - IBM MQ Console è abilitato.
  - Si sta utilizzando il servizio IBM Cloud Pak for Integration Keycloak per l'autorizzazione e l'autenticazione SSO (single sign - on) IBM MQ Console (impostazione predefinita).

  Se tutte le istruzioni precedenti sono vere, SSO viene fornito da Keycloak. Pertanto, oltre a quanto indicato per l' IBM MQ Operator, è necessario ripetere i passaggi per ciascuno di questi componenti aggiuntivi richiesti:
  - IBM Cloud Pak foundational services
  - IBM Cloud Pak for Integration
  - Keycloak (operatoreRed Hat OpenShift)

Per creare immagini di mirroring, consultare Mirroring delle immagini per un cluster air - gapped.

4. Aggiungere l'origine del catalogo IBM MQ Operator .

   Aggiungi l'origine del catalogo che rende IBM MQ Operator disponibile al tuo cluster utilizzando i seguenti valori:

   ```
   export OPERATOR_PACKAGE_NAME=ibm-mq
   export OPERATOR_VERSION=CHOSEN_OPERATOR_VERSION
   export ARCH=ARCHITECTURE
   ```

   Dove

   - *ARCHITECTURE* è l'architettura del sistema su cui si sta distribuendo il file IBM MQ Operator e ha un valore di `amd64`, `ppc64le` o `s390x`.
   - *CHOSEN_OPERATOR_VERSION* è la versione richiesta di IBM MQ Operator. La versione attuale CD è 3.6.0. La versione attuale SC2 è 3.2.13.

   Esistono alcuni componenti aggiuntivi richiesti quando si distribuisce almeno un gestore code in cui **tutte** le seguenti istruzioni sono vere:

   - Si sta utilizzando una licenza CP4I .
   - IBM MQ Console è abilitato.
   - Si sta utilizzando il servizio IBM Cloud Pak for Integration Keycloak per l'autorizzazione e l'autenticazione SSO (single sign - on) IBM MQ Console (impostazione predefinita).

Se tutte le istruzioni precedenti sono vere, SSO viene fornito da Keycloak. Pertanto, oltre che per l'origine del catalogo IBM MQ Operator , è necessario ripetere anche la procedura per ciascuno dei seguenti componenti aggiuntivi richiesti:

- IBM Cloud Pak foundational services
- IBM Cloud Pak for Integration
- Keycloak (operatoreRed Hat OpenShift)

Segui la procedura per le tue origini di catalogo richieste in Aggiunta di origini di catalogo a un cluster.

5. Installare IBM MQ Operator.

   Scegli una delle seguenti due opzioni (utilizza la console o utilizza la CLI):

   - Opzione 1: Installare IBM MQ Operator utilizzando la console OpenShift.
   - Opzione 2: Installa IBM MQ Operator utilizzando la CLI OpenShift.

6. Creare il segreto della chiave di titolarità (solo installazioni online).

   IBM MQ Operator distribuisce le immagini del gestore code estratte da un registro contenitore che esegue un controllo di 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 nello spazio dei nomi in cui verranno installati i gestori code, seguire queste istruzioni per ottenere una chiave di titolarità e creare un segreto di pull.

   Nota: La chiave di titolarità non è richiesta se verranno distribuiti solo i gestori code IBM MQ Advanced for Developers (non Warranted).

   Puoi creare il segreto della chiave di titolarità utilizzando la console OpenShift o la CLI. Il seguente esempio utilizza la CLI:

   a. Ottieni la chiave di titolarità assegnata al tuo ID IBM . 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.
   c. Dalla CLI OpenShift , immetti il seguente comando per creare un segreto di pull dell'immagine denominato `ibm-entitlement-key`.

      ```
      oc create secret docker-registry ibm-entitlement-key \
      --docker-server=cp.icr.io \
      --docker-username=cp \
      --docker-password=entitlement_key \
      --docker-email=user_email
      \--namespace=namespace
      ```

      Dove *entitlement_key* è la chiave di titolarità che hai copiato nel passo b, *user_email>* è l' IBM ID associato al software autorizzato e *namespace* è lo spazio dei nomi in cui hai installato il tuo IBM MQ Operator .

7. Opzionale: Installare CP4I e le relative dipendenze.

   Esistono alcuni componenti aggiuntivi richiesti quando si distribuisce almeno un gestore code in cui **tutte** le seguenti istruzioni sono vere:

   - Si sta utilizzando una licenza CP4I .
   - IBM MQ Console è abilitato.
   - Si sta utilizzando il servizio CP4I Keycloak per l'autorizzazione e l'autenticazione SSO (single sign - on) IBM MQ Console (impostazione predefinita).

   Se tutte le precedenti istruzioni sono true, SSO viene fornito da Keycloak ed è necessario completare le seguenti operazioni aggiuntive:

   - Installare l'operatore IBM Cloud Pak foundational services nella stessa modalità di installazione dell'operatore CP4I . Per le versioni supportate, vedi Versioni del canale operatore per questa release.

- Installare l'operatore CP4I.
- Facoltativo: distribuisci l'IU della piattaforma.

  a. Crea lo spazio dei nomi `ibm-common-services` . Quando accedi al tuo cluster OpenShift tramite la CLI, immetti il seguente comando:

  ```
  oc new-project ibm-common-services
  ```

  b. Distribuisci la IU della piattaforma.

8. Distribuire License Service.

   Ciò è necessario per il monitoraggio dell'utilizzo della licenza dei gestori code. Seguire le istruzioni riportate in Distribuzione del servizio di licenza License Service.

9. Distribuire un gestore code.

   Per istruzioni sulla distribuzione di un gestore code di esempio "avvio rapido" , consultare “Distribuzione di un gestore code semplice utilizzando IBM MQ Operator” a pagina 98.

Attività correlate

“Disinstallazione di IBM MQ Operator” a pagina 84
Puoi utilizzare la console o la CLI Red Hat OpenShift per disinstallare IBM MQ Operator da Red Hat OpenShift.

OpenShift CP4I Aggiornamento di IBM MQ Operator e dei gestori code

Esistono diversi processi di aggiornamento per gli utenti di IBM MQ Operator, a seconda se si utilizzano licenze IBM MQ o licenze IBM Cloud Pak for Integration (CP4I). Completare il passo di aggiornamento per il tipo di distribuzione.

Informazioni su questa attività

Per aggiornare IBM MQ Operator e i gestori code, completare una delle seguenti operazioni:

Procedura

- Opzione 1: **aggiornare le distribuzioni alla versione più recente sul tuo canale Operatore corrente.**

  Per aggiornare le distribuzioni di IBM MQ Operator alla versione più recente sul tuo canale Operatore corrente, vedi “Aggiornamento a un'ultima release di sicurezza del canale IBM MQ Operator” a pagina 70.

- Opzione 2: **Aggiornare le licenze IBM MQ Operator per IBM MQ .**

  Per aggiornare le distribuzioni di IBM MQ Operator in cui vengono utilizzate **solo** IBM MQ licenze, consultare “Aggiornamento di IBM MQ Operator” a pagina 69.

- Opzione 3: **Aggiornare IBM MQ Operator per gli utenti CP4I .**

  Aggiornare le distribuzioni di IBM MQ Operator per utenti di IBM Cloud Pak for Integration. Ciò include se è stato distribuito almeno uno dei gestori code con una licenza CP4I . Consultare “Aggiornamento degli utenti IBM MQ Operator per CP4I” a pagina 79.

OpenShift *Aggiornamento di IBM MQ Operator*

Aggiornare le distribuzioni di IBM MQ Operator in cui vengono utilizzate **solo** IBM MQ licenze.

Prima di iniziare

Importante: Questa attività è riservata agli utenti delle licenze IBM MQ Operator e **solo** IBM MQ . Se si è un utente IBM Cloud Pak for Integration (CP4I) o se è stato distribuito almeno uno dei gestori code utilizzando una licenza CP4I , consultare “Aggiornamento degli utenti IBM MQ Operator per CP4I” a pagina 79.

Informazioni su questa attività

Completare la procedura seguente in base all'aggiornamento necessario.

Nota:

- La versione 3.6.0 del IBM MQ Operator è l'ultima CD release.
- La versione 3.2.13 del IBM MQ Operator è l'ultima SC2 release.

Procedura

- Opzione 1:“Aggiornamento a un'ultima release di sicurezza del canale IBM MQ Operator” a pagina 70
- Opzione 2:“L'aggiornamento al canale 3.2.x SC2 del IBM MQ Operator” a pagina 72
- Opzione 3: “Aggiornamento all'attuale CD canale della IBM MQ Operator” a pagina 74

Aggiornamento a un'ultima release di sicurezza del canale IBM MQ Operator

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

Prima di iniziare

Importante: Questo argomento è per aggiornare le distribuzioni di IBM MQ Operator all'ultima release di sicurezza sul canale della distribuzione. Se non si applica alla propria distribuzione, fare riferimento ai percorsi di aggiornamento alternativi descritti in “Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 69.

Informazioni su questa attività

Esistono diverse procedure, a seconda dell'origine del catalogo utilizzata per distribuire il IBM MQ Operator che si sta aggiornando.

Opzione 1: Origine catalogo specifica per IBM MQ Operator

Una nuova versione 'IBM MQ Operator diventa disponibile in un cluster OpenShift **solo** dopo aver aggiornato l'origine del catalogo. Questo processo fornisce in modo efficace il controllo manuale degli aggiornamenti, quindi non è necessario utilizzare l'opzione **Manuale** per l'impostazione **Update approval** per gli operatori. L'opzione **Manuale** forza l'esecuzione di tutti i possibili aggiornamenti contemporaneamente e può bloccare gli aggiornamenti, quindi utilizzare solo l'opzione **Automatico** .

Per utilizzare questa opzione, passare a Aggiorna con l'origine del catalogo specifica per IBM MQ Operator.

Opzione 2: IBM Operator Catalog

Con questa opzione, le nuove versioni dell'operatore diventano disponibili e vengono applicate **senza** alcun intervento da parte dell'utente. Quindi, utilizzare questa opzione **solo** per le installazioni in linea in cui si desiderano aggiornamenti **automatici** di IBM MQ Operatore in cui non sono necessarie installazioni deterministiche. Questa opzione può essere utile per ambienti di prova, ma **non è adatta per ambienti di produzione**.

Per utilizzare questa opzione, vai a Upgrade with the IBM Operator Catalog.

Per passare dall'utilizzo del Catalogo operatore IBM all'utilizzo dell'origine del catalogo specifico per IBM MQ Operator, che fornisce un maggiore controllo sugli aggiornamenti, consultare “Passaggio all'origine del catalogo specifico per IBM MQ Operator” a pagina 78.

Procedura

- **Aggiornamento con l'origine del catalogo specifica per IBM MQ Operator**

  a) Immagini speculari (solo air-gap).

Se il cluster si trova in un ambiente di rete con restrizioni (air gap), è necessario eseguire il mirroring delle immagini dell' IBM MQ .

Importante: Dalla versione 3.5.0 l' IBM MQ CASE è cambiato e ora contiene una sola versione di queue manager. Il processo di mirroring delle immagini è notevolmente più veloce, ma potrebbe essere necessario eseguirlo più volte, una per ogni versione del gestore di code che si sta installando.

Per ulteriori informazioni, consultare “IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

Devi cercare quali versioni di IBM MQ CASE contengono le versioni di operatore e gestore di coda che ti servono e ripetere il processo di mirroring air-gap per ciascuna di esse. Vedere l' IBM MQ CASE Lookup Tables for the version mappings.

Ad esempio, quando si seguono le istruzioni collegate di seguito per eseguire il mirroring delle immagini, si installa l'ultima versione dell'operatore e della singola versione corrispondente del gestore della coda utilizzando i seguenti valori:

```
export OPERATOR_PACKAGE_NAME=ibm-mq
export OPERATOR_VERSION=3.6.0
```

Per informazioni sul contenuto di IBM MQ CASE, utilizzato per le installazioni e gli aggiornamenti air-gap, vedere '“IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

Per creare immagini speculari, vedere Mirroring delle immagini per un cluster air-gapped.

b) Applica l'ultima origine del catalogo.

Seguire le istruzioni in "Aggiungi origini di catalogo specifiche per IBM MQ Operator" in Aggiunta dell'origine di catalogo IBM MQ Operator.

c) Aggiornare il IBM MQ Operator.

Se lo stato di **approvazione dell'aggiornamento** per il IBM MQ Operator è impostato su **Automatico**, l'operatore si aggiorna. Altrimenti, aggiornare manualmente il file 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 **Aggiorna disponibile**

e. Fare clic su **Anteprima InstallPlan**

f. Fare clic su **Approva** per completare l'aggiornamento

L'operatore esegue l'aggiornamento alla nuova versione.

d) Aggiornare i gestori code IBM MQ .

Seguire le istruzioni riportate in Aggiornamento dei gestori di code IBM MQ.

- **Aggiornamento con il IBM Catalogo operatore**

a) Aggiornare IBM MQ Operator a una versione più recente.

Se si dispone di aggiornamenti automatici impostati, al rilascio di un nuovo rilascio di sicurezza IBM MQ Operator completa un aggiornamento. Altrimenti, approvare manualmente l'aggiornamento IBM MQ Operator

– Se è disponibile un upgrade, **`Upgrade Status`** potrebbe essere "`Upgrade available`".

– In questo caso, potrebbe essere disponibile un controllo che è possibile utilizzare per approvare il **InstallPlan** che aggiorna il IBM MQ Operator.

b) Aggiornare qualsiasi gestore code IBM MQ

Seguire le istruzioni riportate in Aggiornamento dei gestori di code IBM MQ.

- **Aggiorna IBM MQ gestori code**.

  È necessario aggiornare qualsiasi gestore code IBM MQ a una versione più recente dopo l'aggiornamento di IBM MQ Operator. La tabella seguente descrive l'ultima versione del gestore di code IBM MQ per ogni canale Operatore attivo. Utilizzando la versione pertinente, seguire la procedura descritta in “Aggiornamento di un gestore code IBM MQ utilizzando Red Hat OpenShift” a pagina 83.

Canale operatore	Gestore code IBM MQ più recente
3.2SC2)	9.4.0.11-r3
3.6CD)	9.4.3.0-r1

L'aggiornamento al canale 3.2.x SC2 del IBM MQ Operator

Aggiornare le distribuzioni del IBM MQ Operator in cui vengono utilizzate **solo le** licenze IBM MQ

Prima di iniziare

Importante:

- Questo compito è riservato agli utenti delle licenze IBM MQ Operator e **solo** IBM MQ. Se siete un utente IBM Cloud Pak for IntegrationCP4I) o avete distribuito almeno uno dei vostri gestori di code utilizzando una licenza CP4I, consultate “Aggiornamento degli utenti IBM MQ Operator per CP4I” a pagina 79.
- Questo compito serve **solo** per aggiornare le distribuzioni del canale IBM MQ Operator al canale Support Cycle 2SC2) di IBM MQ Operator 3.2.x. Si noti che questo è anche un potenziale passo intermedio quando si aggiorna all'ultima CD release (3.6.0). Se questo non si applica alla vostra installazione, consultate i percorsi di aggiornamento alternativi descritti in “Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 69.

Informazioni su questa attività

Completare una delle seguenti opzioni a seconda della distribuzione del IBM MQ Operator che si sta aggiornando al canale 3.2.x SC2

Procedura

- **Opzione 1**: aggiornare le distribuzioni della versione 2.0.x Long Term SupportLTS) IBM MQ Operator.

  Segui la procedura descritta in “Aggiornamento di un canale 2.0.x LTS IBM MQ Operator al canale 3.2.x SC2” a pagina 72.
- **Opzione 2**: Aggiornare una distribuzione CD del IBM MQ Operator.

  Segui la procedura descritta in “Aggiornamento di un CD IBM MQ Operator al canale v3.2.x SC2 ” a pagina 73.

Aggiornamento di un canale 2.0.x LTS IBM MQ Operator al canale 3.2.x SC2

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

Prima di iniziare

Importante:

- Questa attività è riservata agli utenti delle licenze IBM MQ Operator e **solo** IBM MQ . Se si è un utente IBM Cloud Pak for Integration (CP4I) o se è stato distribuito almeno uno dei gestori code utilizzando una licenza CP4I , consultare “Aggiornamento degli utenti IBM MQ Operator per CP4I” a pagina 79.
- Questo argomento è per l'aggiornamento delle distribuzione di 2.0.x Long Term Support (LTS) IBM MQ Operator al canale Support Cycle 2 (SC2) di IBM MQ Operator 3.2.x **solo**. Se ciò non si applica alla

propria distribuzione, consultare i percorsi di aggiornamento alternativi descritti in “Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 69.

Per eseguire l'aggiornamento a IBM MQ Operator 3.2.13 , è necessario che sia in esecuzione Red Hat OpenShift Container Platform 4.12 o versione successiva. Per verificare le versioni compatibili per ogni canale IBM MQ Operator , consultare “Versioni Red Hat OpenShift Container Platform compatibili” a pagina 36. Per aggiornare la piattaforma, consultare Aggiornamento di Red Hat OpenShift.

Procedura

1. **Immagini speculare (solo air - gap).**

   Se il cluster si trova in un ambiente di rete con restrizioni (air-gapped), è necessario eseguire il mirroring delle immagini dell' IBM MQ .

   Importante: Dalla versione 3.5.0 l' IBM MQ CASE è cambiato e ora contiene una sola versione di queue manager. Il processo di mirroring delle immagini è notevolmente più veloce, ma potrebbe essere necessario eseguirlo più volte, una per ogni versione del gestore di code che si sta installando.

   Per ulteriori informazioni, consultare “IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

   Devi cercare quali versioni di IBM MQ CASE contengono le versioni di operatore e gestore di coda che ti servono e ripetere il processo di mirroring air-gap per ciascuna di esse. Vedere l' IBM MQ CASE Lookup Tables for the version mappings.

   Ad esempio, quando si seguono le istruzioni collegate di seguito per eseguire il mirroring delle immagini, si installa l'ultima versione dell'operatore e la singola versione corrispondente del gestore della coda utilizzando i seguenti valori:

   ```
   export OPERATOR_PACKAGE_NAME=ibm-mq
   export OPERATOR_VERSION=3.6.0
   ```

   Dovresti omettere la sezione 3.5 "Configura il cluster", perché la connessione al registro delle immagini dovrebbe essere stata impostata durante le installazioni o gli aggiornamenti precedenti.

   Link: Mirroring delle immagini per un cluster air - gapped.
2. **Aggiornare IBM MQ Operator a 3.2.13**

   Consultare “Aggiornamento di IBM MQ Operator utilizzando Red Hat OpenShift” a pagina 81.
3. **Aggiorna le istanze.**

   Per ricevere le funzioni e le correzioni di sicurezza più recenti, aggiornare IBM MQ Operando (Immagine Contenitore gestore code) all'ultima versione SC2 (9.4.0.11). Consultare “Aggiornamento di un gestore code IBM MQ utilizzando Red Hat OpenShift” a pagina 83.

Aggiornamento di un CD IBM MQ Operator al canale v3.2.x SC2

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

Prima di iniziare

Importante:

- Questa attività è riservata agli utenti delle licenze IBM MQ Operator e **solo** IBM MQ . Se si è un utente IBM Cloud Pak for Integration (CP4I) o se è stato distribuito almeno uno dei gestori code utilizzando una licenza CP4I , consultare “Aggiornamento degli utenti IBM MQ Operator per CP4I” a pagina 79.
- Questo task serve per aggiornare 'Continuous Delivery (CD) le distribuzioni di 'IBM MQ Operator precedenti alla versione 3.2.0, **solo** alla versione '3.2.13. Si noti che questo è anche un potenziale passo intermedio quando si aggiorna all'ultima release 'CD (v3.6.0). Se questo non si applica alla vostra installazione, consultate i percorsi di aggiornamento alternativi descritti in '“Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 69.

Per eseguire l'aggiornamento a IBM MQ Operator 3.2.13 , è necessario che sia in esecuzione Red Hat OpenShift Container Platform 4.12 o versione successiva. Per verificare le versioni compatibili per ogni canale IBM MQ Operator , consultare “Versioni Red Hat OpenShift Container Platform compatibili” a pagina 36. Per aggiornare la piattaforma, consultare Aggiornamento di Red Hat OpenShift.

Procedura

1. Opzionale: **Aggiornare un IBM MQ Operator che è attualmente alla versione CD precedente a 3.0.0.**

   Se il tuo IBM MQ Operator è attualmente a una versione CD precedente a 3.0.0, segui i passi pertinenti in Migrazione al canale CD corrente dell'operatore IBM MQ (IBM MQ 9.3 documentation), quindi torna qui per eseguire l'aggiornamento all'ultima versione di CD . Notare che questo è un passo prerequisito obbligatorio prima di eseguire l'aggiornamento alla versione 3.2.13.
2. **Aggiornamento all'operatore più recente del vostro canale attuale**

   Segui la procedura descritta in “Aggiornamento a un'ultima release di sicurezza del canale IBM MQ Operator” a pagina 70.
3. **Immagini speculare (solo air - gap).**

   Se il cluster si trova in un ambiente di rete con restrizioni (air-gapped), è necessario eseguire il mirroring delle immagini dell' IBM MQ .

   Importante: Dalla versione 3.5.0 l' IBM MQ CASE è cambiato e ora contiene una sola versione di queue manager. Il processo di mirroring delle immagini è notevolmente più veloce, ma potrebbe essere necessario eseguirlo più volte, una per ogni versione del gestore di code che si sta installando.

   Per ulteriori informazioni, consultare “IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

   Devi cercare quali versioni di IBM MQ CASE contengono le versioni di operatore e gestore di coda che ti servono e ripetere il processo di mirroring air-gap per ciascuna di esse. Vedere l' IBM MQ CASE Lookup Tables for the version mappings.

   Ad esempio, quando si seguono le istruzioni collegate di seguito per eseguire il mirroring delle immagini, si installa l'ultima versione dell'operatore e la singola versione corrispondente del gestore della coda utilizzando i seguenti valori:

   ```
   export OPERATOR_PACKAGE_NAME=ibm-mq
   export OPERATOR_VERSION=3.6.0
   ```

   Dovresti omettere la sezione 3.5 "Configura il cluster", perché la connessione al registro delle immagini dovrebbe essere stata impostata durante le installazioni o gli aggiornamenti precedenti.

   Link: Mirroring delle immagini per un cluster air - gapped.
4. **Aggiornare IBM MQ Operator a 3.2.13**

   Consultare “Aggiornamento di IBM MQ Operator utilizzando Red Hat OpenShift” a pagina 81.
5. **Aggiornare le istanze.**

   Per ricevere le funzioni e le correzioni di sicurezza più recenti, aggiornare IBM MQ Operando (Immagine Contenitore gestore code) all'ultima versione SC2 (9.4.0.11). Consultare “Aggiornamento di un gestore code IBM MQ utilizzando Red Hat OpenShift” a pagina 83.

OpenShift *Aggiornamento all'attuale CD canale della IBM MQ Operator*

Aggiornare le distribuzioni del 'IBM MQ Operator in cui vengono utilizzate **solo le** licenze 'IBM MQ.

Prima di iniziare

Importante:

- Questo compito è riservato agli utenti delle licenze IBM MQ Operator e **solo** IBM MQ. Se siete un utente IBM Cloud Pak for IntegrationCP4I) o avete distribuito almeno uno dei vostri gestori di code utilizzando una licenza CP4I, consultate “Aggiornamento degli utenti IBM MQ Operator per CP4I” a pagina 79.
- Questo argomento riguarda **solo** l'aggiornamento al canale Continuous DeliveryCD) del IBM MQ Operator (3.6). Se questo non si applica alla vostra installazione, consultate i percorsi di aggiornamento alternativi descritti in “Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 69.

Informazioni su questa attività

Completare una delle seguenti opzioni, a seconda della distribuzione del 'IBM MQ Operator che si sta aggiornando al canale CD '3.6:

Procedura

- **Opzione 1:** aggiornamento delle distribuzioni 3.2.x 'SC2 'IBM MQ Operator.

  Segui la procedura descritta in “Aggiornamento di un 'SC2 'IBM MQ Operator '3.2.x al canale 'CD ' attuale” a pagina 75.
- **Opzione 2:** Aggiornare un'implementazione 'CD del 'IBM MQ Operator.

  Segui la procedura descritta in “Aggiornamento di un canale 'CD 'IBM MQ Operator ' al canale 'CD attuale” a pagina 76.

OpenShift *Aggiornamento di un 'SC2 'IBM MQ Operator '3.2.x al canale 'CD ' attuale*

Aggiornare una distribuzione 3.2.x 'SC2 di 'IBM MQ Operator in cui sono utilizzate **solo** licenze 'IBM MQ.

Prima di iniziare

Importante:

- Questa attività è destinata agli utenti delle licenze " IBM MQ Operator e **solo** " IBM MQ ". Se siete un utente 'IBM Cloud Pak for Integration (CP4I) o avete distribuito almeno uno dei vostri gestori di code utilizzando una licenza 'CP4I, consultate '“Aggiornamento degli utenti IBM MQ Operator per CP4I” a pagina 79.
- Questo task serve **solo** per aggiornare una versione 3.2.x 'SC2 del 'IBM MQ Operator all'attuale canale 'Continuous Delivery (CD) del 'IBM MQ Operator (v3.6). Se questo non si applica alla vostra installazione, consultate il percorso di aggiornamento alternativo descritto in '“Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 69.

Per eseguire l'aggiornamento a 'IBM MQ Operator '3.6.0 è necessario eseguire 'Red Hat OpenShift Container Platform '4.12 o un codice successivo. Per verificare le versioni compatibili per ogni canale 'IBM MQ Operator, vedere '“Versioni Red Hat OpenShift Container Platform compatibili” a pagina 36. Per aggiornare la piattaforma, vedere Aggiornamento di 'Red Hat OpenShift nella documentazione di 'IBM Cloud Pak for Integration.

Procedura

1. Opzionale: **Aggiornare un 'IBM MQ Operator che attualmente ha una versione precedente alla 3.2.0.**

   Se il vostro 'IBM MQ Operator ha una versione precedente alla 3.2.0, seguite i passi pertinenti in '“L'aggiornamento al canale 3.2.x SC2 del IBM MQ Operator” a pagina 72, quindi tornate qui per aggiornare alla versione più recente 'CD. Si noti che se si è già su una versione v3.2.x del 'IBM MQ Operator si può saltare questo passaggio, tuttavia l'aggiornamento al canale v3.2 del 'IBM MQ Operator è un passaggio obbligatorio prima dell'aggiornamento alla versione '3.6.0.
2. **Aggiornamento all'operatore più recente del vostro canale attuale**

   Segui la procedura descritta in “Aggiornamento a un'ultima release di sicurezza del canale IBM MQ Operator” a pagina 70.

3. **Immagini speculari (solo air-gap).**

   Se il cluster si trova in un ambiente di rete con restrizioni (air-gapped), è necessario eseguire il mirroring delle immagini dell' IBM MQ .

   Importante: Dalla versione 3.5.0 l' IBM MQ CASE è cambiato e ora contiene una sola versione di queue manager. Il processo di mirroring delle immagini è notevolmente più veloce, ma potrebbe essere necessario eseguirlo più volte, una per ogni versione del gestore di code che si sta installando.

   Per ulteriori informazioni, consultare “IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

   Devi cercare quali versioni di IBM MQ CASE contengono le versioni di operatore e gestore di coda che ti servono e ripetere il processo di mirroring air-gap per ciascuna di esse. Vedere l' IBM MQ CASE Lookup Tables for the version mappings.

   Ad esempio, quando si seguono le istruzioni collegate di seguito per eseguire il mirroring delle immagini, si installa l'ultima versione dell'operatore e la singola versione corrispondente del gestore della coda utilizzando i seguenti valori:

   ```
   export OPERATOR_PACKAGE_NAME=ibm-mq
   export OPERATOR_VERSION=3.6.0
   ```

   Si dovrebbe omettere la sezione 3.5 "Configurazione del cluster", perché la connessione al registro delle immagini dovrebbe essere stata impostata durante le installazioni o gli aggiornamenti precedenti.

   Per informazioni sul contenuto di IBM MQ CASE, utilizzato per le installazioni e gli aggiornamenti air-gap, vedere '“IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

   Collegamento: Immagini di mirroring per un cluster con gapsula ad aria.
4. **Aggiornare il 'IBM MQ Operator a '3.6.0.**

   Seguire le istruzioni in “Aggiornamento di IBM MQ Operator utilizzando Red Hat OpenShift” a pagina 81.
5. **Aggiornare le istanze.**

   Per ricevere le ultime funzionalità e le correzioni di sicurezza, aggiornare l'Operand IBM MQ (immagine del Queue Manager Container) alla versione più recente CD (9.4.3.0-r1). Vedere '“Aggiornamento di un gestore code IBM MQ utilizzando Red Hat OpenShift” a pagina 83.

OpenShift *Aggiornamento di un canale 'CD 'IBM MQ Operator ' al canale 'CD attuale*

Aggiornare una distribuzione 'CD di 'IBM MQ Operator in cui vengono utilizzate **solo le** licenze 'IBM MQ.

Prima di iniziare

Importante:

- Questo compito è destinato agli utenti delle licenze " IBM MQ Operator e **solo** " IBM MQ ". Se siete un utente 'IBM Cloud Pak for Integration (CP4I) o avete distribuito almeno uno dei vostri gestori di code utilizzando una licenza 'CP4I, consultate '“Aggiornamento degli utenti IBM MQ Operator per CP4I” a pagina 79.
- Questo task serve **solo** per aggiornare una versione 'Continuous Delivery (CD) del 'IBM MQ Operator all'attuale canale 'CD del 'IBM MQ Operator (v3.6). Se questo non è applicabile alla vostra installazione, consultate il percorso di aggiornamento alternativo descritto in '“Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 69.

Per eseguire l'aggiornamento a 'IBM MQ Operator '3.6.0 è necessario eseguire 'Red Hat OpenShift Container Platform '4.12 o un codice successivo. Per verificare le versioni compatibili per ogni canale 'IBM MQ Operator, vedere '“Versioni Red Hat OpenShift Container Platform compatibili” a pagina 36. Per aggiornare la piattaforma, vedere Aggiornamento di 'Red Hat OpenShift nella documentazione di 'IBM Cloud Pak for Integration.

Procedura

1. Opzionale: **Aggiornare un 'IBM MQ Operator che attualmente ha una versione precedente alla 3.2.0.**

   Se il vostro 'IBM MQ Operator ha una versione precedente alla 3.2.0, seguite i passi pertinenti in '"L'aggiornamento al canale 3.2.x SC2 del IBM MQ Operator" a pagina 72, quindi tornate qui per seguire il passo '"2" a pagina 77 per aggiornare alla versione più recente 'CD. Si noti che se si è già su una versione '3.2.x del 'IBM MQ Operator si può saltare questo passaggio, ma l'aggiornamento al canale v3.2 del 'IBM MQ Operator è un passaggio obbligatorio prima dell'aggiornamento alla versione '3.6.0.

2. Opzionale: **Aggiornare un 'IBM MQ Operator che attualmente è alla versione '3.2.x.**

   Se il vostro 'IBM MQ Operator è attualmente alla versione '3.2.x, seguite i passi relativi al '"Aggiornamento di un 'SC2 'IBM MQ Operator '3.2.x al canale 'CD ' attuale" a pagina 75.

3. **Aggiornare un' IBM MQ Operator e che attualmente è alla versione 3.3.0, o superiore.**

   Se l' IBM MQ Operator e è attualmente alla versione 3.3.0 o superiore, completare i seguenti passaggi per eseguire l'aggiornamento a IBM MQ Operator 3.6.0.

   a) **Aggiornamento all'operatore più recente del vostro canale attuale**

      Segui la procedura descritta in "Aggiornamento a un'ultima release di sicurezza del canale IBM MQ Operator" a pagina 70.

   b) **Passa al canale operatore successivo.**

      i) Immagini speculari (solo traferro).

      Se il cluster si trova in un ambiente di rete con restrizioni (air-gapped), è necessario eseguire il mirroring delle immagini dell' IBM MQ . Completate i passaggi al seguente link, utilizzando solo questi valori:

      ```
      export OPERATOR_PACKAGE_NAME=ibm-mq
      export OPERATOR_VERSION=[operator_version]
      ```

      dove *[operator_version]* è la versione dell' IBM MQ Operator e a cui si sta effettuando l'aggiornamento. Per determinare le ultime informazioni (IBM MQ Operator) sul canale a cui si sta effettuando l'upgrade, vedere "Cronologia delle release per IBM MQ Operator" a pagina 5.

      Si dovrebbe omettere la sezione 3.5 "Configurazione del cluster", perché la connessione al registro delle immagini dovrebbe essere stata impostata durante le installazioni o gli aggiornamenti precedenti.

      Per informazioni sul contenuto di IBM MQ CASE, utilizzato per le installazioni e gli aggiornamenti air-gap, vedere '"IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift" a pagina 295.

      Collegamento: Immagini di mirroring per un cluster con gapsula ad aria.

      ii) Aggiorna il tuo IBM MQ Operator.

      Seguire le istruzioni in "Aggiornamento di IBM MQ Operator utilizzando Red Hat OpenShift" a pagina 81.

      iii) Aggiornare le istanze.

      Per ricevere le ultime funzionalità e correzioni di sicurezza, aggiornare l' IBM MQ . Operand (immagine contenitore del gestore di coda). Vedere "Aggiornamento di un gestore code IBM MQ utilizzando Red Hat OpenShift" a pagina 83.

   c) Ripetere i passaggi precedenti (a) e (b) fino a raggiungere l'ultimo canale di aggiornamento (CD) (v3.6.0).

OpenShift *Passaggio all'origine del catalogo specifico per IBM MQ Operator*

Se si dispone di un'installazione di IBM MQ Operator da una release precedente e si sta utilizzando IBM Operator Catalog, l'applicazione dell'origine del catalogo specifica è il modo più efficace per controllare completamente la versione del software su un cluster.

Prima di iniziare

Importante: Questa attività deve essere eseguita dall'amministratore del cluster. Vedi Ruoli e autorizzazioniOpenShift.

I seguenti passi vengono completati utilizzando la CLI.

Informazioni su questa attività

Il IBM Operator Catalog è un indice di operatori disponibili per estendere l'API di un cluster Red Hat OpenShift Container Platform per abilitare i prodotti software IBM .

Questa procedura sposta un'installazione di IBM MQ Operator dal catalogo operatore IBM in modo da poter utilizzare l'origine del catalogo specifica per IBM MQ Operator.

Procedura

1. Aggiungi il catalogo IBM MQ Operator .

   Seguire le istruzioni in "Aggiungi origini di catalogo specifiche per IBM MQ Operator" in Aggiunta dell'origine di catalogo IBM MQ Operator.

2. Conferma che l'origine del catalogo IBM MQ Operator è stata creata nel namespace `openshift-marketplace` .

   Esegui il seguente comando:

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

   Output di esempio:

   ```
   oc get catalogsource -n openshift-marketplace

   NAME                          DISPLAY               TYPE   PUBLISHER   AGE
   ibm-operator-catalog          IBM Operator Catalog  grpc   IBM         23h
   ibmmq-operator-catalogsource  ibm-mq-3.1.3          grpc   IBM         23h
   ```

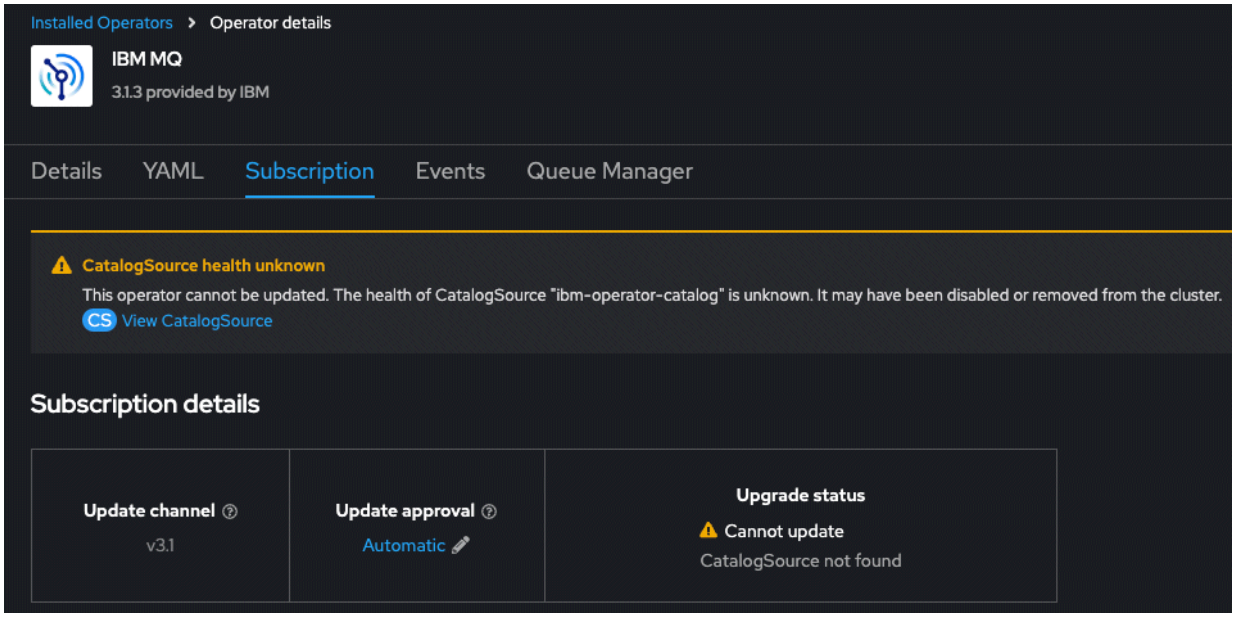
3. Opzionale: Eliminare l'origine del catalogo operatore IBM .

   Avvertenza: È necessario completare questo passo solo se si è certi che non ci sono altri operatori che utilizzano il catalogo degli operatori IBM .

   Esegui il seguente comando:

   ```
   oc delete catalogsource ibm-operator-catalog -n openshift-marketplace
   ```

   Lo stato di IBM MQ Operator cambia in `CatalogSource not found`. Ciò è previsto.



4. Modificare la sottoscrizione di IBM MQ Operator in modo che punti alla nuova origine di catalogo IBM MQ Operator specifica.

   a) Modificare la sottoscrizione.

      Immetti il seguente comando, sostituendo *OPERATOR - NAMESPACE* con `openshift-operators` per le installazioni a livello di cluster di IBM MQ Operatoro con lo spazio dei nomi specifico in cui è distribuito IBM MQ Operator :

      ```
      oc edit subscription ibm-mq -n OPERATOR-NAMESPACE
      ```

   b) Modificare il valore `spec.source` da `ibm-operator-catalog` al nome dell'origine del catalogo creata nel passo “1” a pagina 78.

      Ad esempio:

      ```
      spec:
        channel: v3.1
        installPlanApproval: Automatic
        name: ibm-mq
        source: ibm-operator-catalog # CHANGE --> ibmmq-operator-catalogsource
        sourceNamespace: openshift-marketplace
      ```

   c) Salvare le modifiche.

      L'installazione di IBM MQ Operator ora punta all'origine del catalogo IBM MQ Operator . Se è stato eliminato il catalogo IBM Operator, lo stato passa da "CatalogSource not found" a "Succeeded".

Risultati

L'installazione di IBM MQ Operator ora punta all'origine del catalogo specifico per IBM MQ Operator. Questo ti dà il pieno controllo sugli aggiornamenti per l'operatore.

OpenShift CP4I *Aggiornamento degli utenti IBM MQ Operator per CP4I*

Aggiornare le distribuzioni di IBM MQ Operator in cui viene utilizzata una licenza IBM Cloud Pak for Integration (CP4I).

Prima di iniziare

Importante: Questa attività è per gli utenti CP4I . Ciò include se è stato distribuito almeno uno dei gestori code con una licenza CP4I . Se questo non si applica all'utente, consultare “Aggiornamento di IBM MQ Operator” a pagina 69.

Informazioni su questa attività

Se si sta eseguendo l'aggiornamento alla versione CD del IBM MQ Operator (versione 3.6.0), è necessario prima eseguire l'aggiornamento al canale v3.2 SC2 come descritto nelle istruzioni seguenti.

Se il tuo IBM MQ Operator è versione 3.3.0 o superiore, vai al sotto-passo (c) nel passo `Upgrade to the CD Channel` .

Procedura

1. Opzionale: **Aggiornamento al canale v3.2 SC2.**

   Completate una o più delle seguenti opzioni per aggiornare il vostro IBM MQ Operator al canale Support Cycle 2SC2) del IBM MQ Operator **solo:**

   - **Opzione 1:** aggiornare le distribuzioni della versione 2.0.x Long Term SupportLTS) IBM MQ Operator.

     Segui la procedura in Aggiornamento da 2022.2 generando un piano di upgrade.
   - **Opzione 2:** aggiornare una distribuzione 3.0.x o 3.1.x di IBM MQ Operator.

     Attieniti alla procedura in Upgrading from 2023.4 generando un piano di upgrade.
   - **Opzione 3:** aggiornare altre distribuzioni di IBM MQ Operator.

     Seguire i passaggi pertinenti in Migrazione al canale CD corrente della documentazione IBM MQ Operator (IBM MQ 9.3), quindi tornare qui e procedere con l'**Opzione 2**. Si noti che questo è un passo prerequisito obbligatorio.
2. **Aggiornamento al canale CD (3.6).**

   a) Opzionale: **Passa alla distribuzione dell' 3.2.x IBM MQ Operator**

   Seguire il passo principale precedente per aggiornare alla distribuzione 3.2.x del 'IBM MQ Operator.

   b) Opzionale: **Aggiornare una distribuzione 3.2.x di 'IBM MQ Operator**

   Seguire i passaggi in Aggiornamento da 16.1.0 generando un piano di aggiornamento.

   c) Opzionale: **Aggiornare una distribuzione 3.3.x del sistema IBM MQ Operator**

   Se il vostro IBM MQ Operator è attualmente alla versione 3.3.x, completate i seguenti passi per aggiornare il vostro operatore a 3.4.x prima di continuare con l'aggiornamento successivo a 3.6.0.

   i) Immagini speculari (solo traferro).

   Se il cluster si trova in un ambiente di rete con restrizioni (air-gapped), è necessario eseguire il mirroring delle immagini dell' IBM MQ . Completa i passaggi al seguente link, utilizzando solo questi valori:

   ```
   export OPERATOR_PACKAGE_NAME=ibm-mq
   export OPERATOR_VERSION=3.4.1
   ```

   Dovresti omettere la sezione "Configurazione del cluster" (3.5), perché la connessione al registro immagini dovrebbe essere stata impostata durante le precedenti installazioni o aggiornamenti.

   Per informazioni sul contenuto dell' IBM MQ CASE, che viene utilizzato per installazioni e aggiornamenti air-gap, vedere “IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

   Link : Immagini speculari per un cluster air-gapped.

   ii) Aggiorna il tuo IBM MQ Operator.

   Seguire le istruzioni in “Aggiornamento di IBM MQ Operator utilizzando Red Hat OpenShift” a pagina 81.

   iii) Aggiornare le istanze.

Per ricevere le ultime funzionalità e correzioni di sicurezza, aggiornare l' IBM MQ . Operand (immagine contenitore del gestore di coda). Vedere “Aggiornamento di un gestore code IBM MQ utilizzando Red Hat OpenShift” a pagina 83.

d) **Aggiornare una distribuzione 3.4.0 o superiore del sistema IBM MQ Operator**

i) Immagini speculari (solo traferro).

Se il cluster si trova in un ambiente di rete con restrizioni (air-gapped), è necessario eseguire il mirroring delle immagini dell' IBM MQ . Completa i passaggi al seguente link, utilizzando solo questi valori:

```
export OPERATOR_PACKAGE_NAME=ibm-mq
export OPERATOR_VERSION=[operator_version]
```

dove *[operator_version]* è la versione dell' IBM MQ Operator e a cui si sta effettuando l'aggiornamento. Per determinare le ultime informazioni (IBM MQ Operator) sul canale a cui si sta effettuando l'aggiornamento, vedere “Cronologia delle release per IBM MQ Operator” a pagina 5.

Dovresti omettere la sezione "Configurazione del cluster" (3.5), perché la connessione al registro immagini dovrebbe essere stata impostata durante le precedenti installazioni o aggiornamenti.

Per informazioni sul contenuto dell' IBM MQ CASE, che viene utilizzato per installazioni e aggiornamenti air-gap, vedere “IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift” a pagina 295.

Link : Immagini speculari per un cluster air-gapped.

ii) Passare all'operatore più recente del canale attuale.

Segui la procedura descritta in “Aggiornamento a un'ultima release di sicurezza del canale IBM MQ Operator” a pagina 70.

iii) Aggiornare la distribuzione IBM MQ Operator .

Seguire la procedura descritta in Aggiornamento da 16.1.1 generando un piano di aggiornamento.

OpenShift CP4I *Aggiornamento di IBM MQ Operator utilizzando Red Hat OpenShift*

Il 'IBM MQ Operator può essere aggiornato dalla console o dalla riga di comando del 'Red Hat OpenShift.

Prima di iniziare

Nota:

- L'ultima CD versione del IBM MQ Operator è 3.6.0.
- L'ultima SC2 versione del IBM MQ Operator è 3.2.13.

Per le ultime novità IBM MQ Operator note di rilascio, vedere Cronologia delle versioni per IBM MQ Operator.

Procedura

- **Opzione 1: Aggiornare il 'IBM MQ Operator utilizzando la console 'Red Hat OpenShift.**
  a) Accedere alla console del cluster 'Red Hat OpenShift.
  b) Immagini speculari (solo air-gap).

Se il cluster si trova in un ambiente di rete limitato (air-gapped), è necessario eseguire il mirroring delle immagini 'IBM MQ utilizzando i seguenti valori:

```
export OPERATOR_PACKAGE_NAME=ibm-mq
export OPERATOR_VERSION=3.6.0
```

Per informazioni sul contenuto di IBM MQ CASE, utilizzato per le installazioni e gli aggiornamenti air-gap, vedere '"IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift" a pagina 295.

Per creare immagini speculari, vedere Mirroring delle immagini per un cluster air-gapped.

c) Applicare l'ultima fonte del catalogo.

Se si utilizza l'origine catalogo specifica per il 'IBM MQ Operator, anziché per il '`ibm-operator-catalog`, è necessario applicare l'origine catalogo per la nuova versione 'IBM MQ.

Per passare dall'uso del catalogo operatori 'IBM all'uso della fonte di catalogo specifica per il 'IBM MQ Operator e ottenere un maggiore controllo sugli aggiornamenti, consultare i passi del '"Passaggio all'origine del catalogo specifico per IBM MQ Operator" a pagina 78 prima di tornare a completare il passo successivo.

Se si utilizza il catalogo operatori 'IBM (solo per alcune installazioni online), passare al punto successivo.

Altrimenti, seguite le istruzioni del " "Aggiunta dell'origine del catalogo IBM MQ Operator" a pagina 60 e passate al passo successivo.

d) Aggiornare il 'IBM MQ Operator. Le nuove versioni maggiori/minori del " IBM MQ Operator vengono fornite attraverso nuovi canali di abbonamento. Per aggiornare l'Operatore a una nuova versione maggiore/minore, è necessario aggiornare il canale selezionato nella Sottoscrizione 'IBM MQ Operator.

a. Nel riquadro di navigazione, fare clic su **OperatoriOperatori** > **installati**.

Vengono visualizzati tutti gli operatori installati nel progetto specificato.

b. Selezionare l'**operatoreIBM MQ**.

c. Passare alla scheda **Abbonamento**.

d. Fare clic sul **Canale**.

Viene visualizzata la finestra **Modifica canale di aggiornamento della sottoscrizione**.

e. Selezionare il canale desiderato, quindi fare clic su **Salva**.

L'operatore esegue l'aggiornamento all'ultima versione disponibile per il nuovo canale. Consultare "Supporto versione per IBM MQ Operator" a pagina 35.

- **Opzione 2: Aggiornare il 'IBM MQ Operator utilizzando la riga di comando 'Red Hat OpenShift.**

a) Accedere al cluster utilizzando '**`oc login`**.

b) Immagini speculari (solo air-gap).

Se il cluster si trova in un ambiente di rete limitato (air-gapped), è necessario eseguire il mirroring delle immagini 'IBM MQ utilizzando i seguenti valori:

```
export OPERATOR_PACKAGE_NAME=ibm-mq
export OPERATOR_VERSION=3.6.0
```

Per informazioni sul contenuto di IBM MQ CASE, utilizzato per le installazioni e gli aggiornamenti air-gap, vedere '"IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift" a pagina 295.

Per creare immagini speculari, vedere Mirroring delle immagini per un cluster air-gapped.

c) Applicare l'ultima fonte del catalogo.

Se si utilizza l'origine catalogo specifica per il 'IBM MQ Operator, anziché per il '`ibm-operator-catalog`, è necessario applicare l'origine catalogo per la nuova versione 'IBM MQ.

Per passare dall'uso del catalogo operatori 'IBM all'uso della fonte di catalogo specifica per il 'IBM MQ Operator e ottenere un maggiore controllo sugli aggiornamenti, consultare i passi del '"Passaggio all'origine del catalogo specifico per IBM MQ Operator" a pagina 78 prima di tornare a completare il passo successivo.

Se si utilizza il catalogo operatori 'IBM (solo per alcune installazioni online), passare al punto successivo.

Altrimenti, seguite le istruzioni del " "Aggiunta dell'origine del catalogo IBM MQ Operator" a pagina 60 e passate al passo successivo.

d) Aggiornare il 'IBM MQ Operator. Le nuove versioni maggiori o minori del " IBM MQ Operator vengono fornite attraverso nuovi canali di abbonamento. Per aggiornare l'Operatore a una nuova versione maggiore o minore, completare i passaggi seguenti per aggiornare il canale selezionato nella Sottoscrizione 'IBM MQ Operator.

a. Assicurarsi che sia disponibile il canale di aggiornamento 'IBM MQ Operator richiesto.

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

b. Applicare il 'Subscription per spostarsi sul canale di aggiornamento desiderato (dove *vX.Y* è il canale di aggiornamento desiderato identificato nel passaggio precedente).

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

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

Un IBM MQ Queue Manager, distribuito utilizzando il IBM MQ Operator, può essere aggiornato in Red Hat OpenShift utilizzando una delle seguenti interfacce: Hub operatore; riga di comando; IBM Cloud Pak for Integration Platform UI.

Prima di iniziare

Come parte del processo di aggiornamento dei gestori code IBM MQ , è possibile che l'utente sia stato inviato a questo argomento dalla documentazione IBM Cloud Pak for Integration .

Nota:

- L'ultima CD versione del IBM MQ gestore di code è 9.4.3.0-r1.
- L'ultima SC2 versione del IBM MQ gestore di code è 9.4.0.11-r3.

Per le ultime IBM MQ note di rilascio del gestore di code, vedere "Cronologia dei rilasci per le immagini dei container di queue manager" a pagina 18.

Assicuratevi che il vostro IBM MQ Operator sia alla versione richiesta. Per aggiornare il proprio IBM MQ Operator, vedere "Aggiornamento di IBM MQ Operator" a pagina 69.

Prima di poter aggiornare il gestore di code in un ambiente air-gap, è necessario eseguire il mirroring delle immagini più recenti IBM Cloud Pak for Integration attraverso i passaggi specifici per l'air-gap che vengono completati durante l'aggiornamento del IBM MQ Operator. Per un esempio di questi passaggi, vedere "Aggiornamento all'attuale CD canale della IBM MQ Operator" a pagina 74.

Informazioni su questa attività

Questo task offre tre opzioni per l'aggiornamento di un IBM MQ Queue Manager distribuito su OpenShift:

1. Aggiornamento di un IBM MQ gestore di code utilizzando la Red Hat OpenShift console.
2. Aggiornamento di un IBM MQ gestore di code utilizzando la Red Hat OpenShift CLI
3. Aggiornamento di un IBM MQ gestore di code in Red Hat OpenShift utilizzando l'interfaccia utente della piattaforma

Procedura

- **Opzione 1**: Aggiornamento di un IBM MQ gestore di code utilizzando la Red Hat OpenShift console.
  a) Accedere alla console web del cluster Red Hat OpenShift
  b) Nel riquadro di navigazione, fare clic su **OperatoriOperatori** > **installati**.
  Vengono visualizzati tutti gli operatori installati nel progetto specificato.
  c) Selezionare l'**operatoreIBM MQ**.
  Viene visualizzata la finestra **IBM MQ Operator**.
  d) Passare alla scheda **Queue Manager**.
  Viene visualizzata la finestra **Dettagli Queue Manager**.
  e) Selezionare il gestore di code che si desidera aggiornare.
  f) Passare alla scheda YAML.
  g) Aggiornare i seguenti campi, se necessario, in modo che corrispondano all'aggiornamento della versione di IBM MQ queue manager desiderato.
  – spec.version
  – spec.license.licence
  Vedere "Cronologia dei rilasci per le immagini dei container di queue manager" a pagina 18 per una mappatura delle IBM MQ Operator versioni e delle IBM MQ immagini dei contenitori dei gestori di code.
  h) Salvare lo YAML del gestore di code aggiornato.
- **Opzione 2**: Aggiornamento di un IBM MQ gestore di code utilizzando la Red Hat OpenShift CLI
  a) Log in to the Red Hat OpenShift command line interface (CLI) using `oc login`.
  b) Modificare la **`QueueManager`** risorsa per aggiornare i seguenti campi, se necessario, in modo che corrispondano all'aggiornamento della IBM MQ versione di queue manager desiderato.
  – spec.version
  – spec.license.licence
  Vedere "Supporto versione per IBM MQ Operator" a pagina 35 per una mappatura dei canali alle IBM MQ Operator versioni e alle IBM MQ versioni dei gestori di code.
  Utilizzare il seguente comando:
  ```
  oc edit queuemanager my_qmgr
  ```
  dove *my_qmgr* è il nome della risorsa QueueManager che si vuole aggiornare.
- **Opzione 3**: aggiornamento di un IBM MQ gestore di code in Red Hat OpenShift utilizzando l'interfaccia utente della piattaforma.
  Seguire le istruzioni pertinenti in Gestione delle versioni e degli aggiornamenti tramite l'interfaccia utente della piattaforma nella documentazione IBM Cloud Pak for Integration.

OpenShift CP4I Disinstallazione di IBM MQ Operator

Puoi utilizzare la console o la CLI Red Hat OpenShift per disinstallare IBM MQ Operator da Red Hat OpenShift.

Procedura

- Opzione 1: disinstallazione di IBM MQ Operator con la console OpenShift .
  Nota: Se IBM MQ Operator è installato su tutti i progetti / namespace sul cluster, ripetere i passi da 2 a 6 della seguente procedura per ogni progetto in cui si desidera eliminare i gestori code.

a) Accedi alla console web Red Hat OpenShift Container Platform con le credenziali di amministratore del cluster Red Hat OpenShift Container Platform .

b) Modificare **Progetto** nello spazio dei nomi da cui si desidera disinstallare IBM MQ Operator. Selezionare lo spazio dei nomi dall'elenco a discesa **Progetto** .

c) Nel riquadro di navigazione, fare clic su **Operatori** > **Operatori installati**.

d) Fare clic sull'operatore **IBM MQ** .

e) Selezionare la scheda **Gestori code** per visualizzare i gestori code gestiti da questo IBM MQ Operator.

f) 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.

g) Opzionale: Se appropriato, ripetere i passi da 2 a 6 per ogni progetto in cui si desidera eliminare i gestori code.

h) Tornare a **Operatori** > **Operatori installati**.

i) Accanto all'operatore **IBM MQ** , fare clic sul menu a tre punti e selezionare **Disinstalla operatore**.

- Opzione 2: disinstallare IBM MQ Operator con la CLI OpenShift

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

b) Se IBM MQ Operator è installato in un singolo spazio dei nomi, completare i seguenti passi secondari:

a. Assicurarsi di trovarsi nel progetto contenente il IBM MQ Operator da disinstallare:

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

i) Disinstallare l'operatore dei servizi comuni, seguendo le istruzioni in Disinstallazione dei servizi di base nella documentazione del prodotto IBM Cloud Pak foundational services .

ii) Visualizzare il gruppo di operatori:

```
oc get operatorgroup
```

iii) Eliminare il gruppo di operatori:

```
oc delete OperatorGroup operator_group_name
```

c) 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. Facoltativo: se nessun altro utilizza i servizi comuni, è possibile disinstallare l'operatore dei servizi comuni. Per eseguire questa operazione, seguire le istruzioni in Disinstallazione dei servizi di base nella documentazione del prodotto IBM Cloud Pak foundational services .

V 9.4.1 Kubernetes Preparare l'uso di 'Kubernetes creando un segreto di pull

L'immagine precostruita 'IBM MQ Advanced può essere estratta da 'IBM Container Registry, ma il registro esegue un controllo dei diritti di licenza. Questo controllo richiede una chiave di abilitazione memorizzata in un pull secret " `docker-registry`.

Prima di iniziare

Installare lo strumento a riga di comando '**kubectl** e accedere al cluster 'Kubernetes.

Procedura

1. Ottenere la chiave di abilitazione assegnata all'ID " IBM.
   a) Accedere a MyIBM Container Software Library con l'ID e la password 'IBM associati al software in questione.
   b) Nella sezione **Chiavi di abilitazione**, selezionare **Copia chiave per** copiare la chiave di abilitazione negli appunti.
2. Creare il segreto della chiave di abilitazione.

a. Dalla CLI 'OpenShift, eseguire il comando seguente per creare un'immagine pull secret chiamata 'ibm-entitlement-key.

```
kubectl create secret docker-registry ibm-entitlement-key \
--docker-server=cp.icr.io \
--docker-username=cp \
--docker-password=entitlement_key \
--docker-email=user_email \
--namespace=namespace
```

Dove *entitlement_key* è la chiave di abilitazione copiata al punto b, *user_email* è l'ID 'IBM associato al software abilitato e *namespace* è lo spazio dei nomi che si intende utilizzare per i gestori delle code.

Kubernetes Prepararsi all'uso di Kubernetes installando il programma IBM License Service

Le licenze per i container consentono di concedere in licenza solo la capacità disponibile dei singoli container IBM MQ , invece di richiedere la licenza per l'intero server in cui sono in esecuzione i container. Per sfruttare le licenze per container, è necessario utilizzare il sito IBM License Service per tenere traccia dell'utilizzo delle licenze e determinare i diritti necessari.

Procedura

Installare IBM License Service seguendo le istruzioni riportate in Installare License Service sul cluster Kubernetes nella documentazione IBM Cloud Pak foundational services .

V 9.4.1 Linux Preparazione all'uso di 'Podman o 'Docker mediante l'estrazione dell'immagine del contenitore precostruito

L'immagine precostituita 'IBM MQ Advanced può essere estratta da 'IBM Container Registry utilizzando strumenti a riga di comando, come 'Podman o 'Docker.

Prima di iniziare

Installare 'Podman o 'Docker.

Nota: I seguenti esempi di comandi utilizzano 'Podman, ma è possibile utilizzare 'Podman o 'Docker con gli stessi parametri di comando.

Procedura

1. Ottenere la chiave di abilitazione assegnata al proprio ID IBM.
   a) Accedere a MyIBM Container Software Library con l'ID e la password IBM associati al software autorizzato.
   b) Nella sezione **Tasti di abilitazione**, selezionare **Copia chiave** per copiare la chiave di abilitazione negli appunti.
2. Accedere al registro del contenitore

   Ad esempio, utilizzando 'Podman:

   ```
   podman login --username cp --password ... cp.icr.io
   ```

   Dove "..." viene sostituita con la chiave di abilitazione copiata nel passaggio precedente.
3. Estrarre l'immagine

   Ad esempio, utilizzando 'Podman:

   ```
   podman pull cp.icr.io/cp/ibm-mqadvanced-server:9.4.3.0-r1
   ```

Preparazione per IBM MQ creando la propria immagine contenitore

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 IBM MQ Operator. Vedere “Scelta della modalità di utilizzo di IBM MQ nei contenitori” a pagina 28

Informazioni su questa attività

Procedura

Considerazioni generali quando si crea la propria immagine del gestore code

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.

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 89
  - “Creazione di un'immagine del gestore code IBM MQ configurata di esempio” a pagina 89

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.

  Segui le istruzioni riportate in Creazione di un'immagine del contenitore su GitHub.
- Opzionale: Se si prevede di configurare l'accesso sicuro utilizzando l'SCC (Security Context Constraint) Red Hat OpenShift Container Platform "limitato" , utilizzare una delle immagini di non installazione IBM MQ .

  I link per scaricare queste immagini sono disponibili nella sezione Contenitori di IBM MQ download.

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.

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

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. Nella versione corrente di IBM MQ, quando si utilizza il package "Nessuna installazione" 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}`** .

Esecuzione di applicazioni di bind locali in contenitori separati

Con la condivisione dello spazio dei nomi del processo tra i contenitori, 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

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` .

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 92 per ulteriori informazioni.

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 91 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, 2025.
```

```
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
```

MQ Adv. Kubernetes 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.

Nota: Puoi creare un gruppo HA nativo nei tuoi propri contenitori solo in questo modo se stai eseguendo Kubernetes o Red Hat OpenShift.

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)
```

MQ Adv. V 9.4.2 Kubernetes Creazione di una configurazione nativa HA CRR se si creano i propri container

È necessario creare, configurare e avviare tre gestori di code per creare il gruppo HA nativo.

Informazioni su questa attività

Il metodo consigliato per creare una soluzione Native HA CRR è quello di utilizzare l'operatore IBM MQ (vedi “Configurazione di Native HA CRR utilizzando il IBM MQ Operator” a pagina 115). In alternativa, se crei i tuoi contenitori, puoi seguire queste istruzioni.

Per creare una configurazione Native HA CRR, è necessario creare due gruppi Native HA per un gestore di code, uno sul sito "live" e uno sul sito "di ripristino". Quindi si modifica il file " `qm.ini` " per consentire al gestore della coda di impostare la replica tra le regioni.

Nota: È possibile creare una configurazione Native HA CRR nei propri container in questo modo solo se si utilizza un Kubernetes o un Red Hat OpenShift.

Procedura

1. Sul sito Live, creare un gruppo Native HA, vedere “Creazione del gruppo HA nativo se si creano i propri contenitori” a pagina 93.
2. Sul sito di ripristino, creare un secondo gruppo Native HA, vedere “Creazione del gruppo HA nativo se si creano i propri contenitori” a pagina 93.
3. Specificare che i due gruppi nei siti live e di ripristino lavorano insieme in una configurazione Native HA CRR apportando le seguenti modifiche al file qm.ini per il gestore della coda.

   Nel gruppo di recupero, specificare i seguenti valori nella sezione " `NativeHALocalInstance` ":

   `GroupName`
   Specificare un nome per il gruppo di ripristino, ad esempio "NHACRR_recovery".

   `GroupRole`
   Specificare il ruolo attuale del gruppo. In questo caso, impostare su "Recupero".

   Specificare i seguenti valori nella sezione " `NativeHARecoveryGroup` ":

   `GroupName`
   Il nome del gruppo sul sito live, ad esempio, "NHACRR_live".

   `ReplicationAddress`

   Specificare un nome host e una porta che indirizzi la porta del traffico di replica sull'altro gruppo nella configurazione di replica interregionale Native HA. L'indirizzo può essere un singolo indirizzo di servizio e porta o un elenco separato da virgole.

   Sul gruppo live, specificare i seguenti valori nella strofa dell' `NativeHALocalInstance` :

   `GroupName`
   Specificare un nome per il gruppo live, ad esempio "NHACRR_live".

   `GroupRole`
   Specificare il ruolo attuale del gruppo. In questo caso impostato su "Live".

   Specificare i seguenti valori nella sezione " `NativeHARecoveryGroup` ":

   `GroupName`
   Il nome del gruppo sul sito di recupero, ad esempio, "NHACRR_recovery".

   `ReplicationAddress`

   Specificare un nome host e una porta che indirizzi la porta del traffico di replica sull'altro gruppo nella configurazione di replica interregionale Native HA. L'indirizzo può essere un singolo indirizzo di servizio e porta o un elenco separato da virgole.
4. È necessario proteggere la connessione utilizzata per la replica utilizzando TLS. Se hai specificato la sicurezza per la replica interna quando hai creato i due gruppi, puoi utilizzare le stesse impostazioni per proteggere la replica esterna. In alternativa, è possibile specificare i seguenti valori nella sezione " `NativeHALocalInstance` " per il gruppo di ripristino e il gruppo live.

   `GroupCipherSpec`
   Sostituisce le specifiche di cifratura specificate da `CipherSpec`.

   `GroupCertificateLabel`
   Sostituisce l'etichetta del certificato specificata da `CertificateLabel`.

   Se si desidera utilizzare questi valori per la replica esterna, mentre si utilizza il testo normale per la replica interna, impostare questi due valori per i gruppi live e di ripristino, impostando `CipherSpec` su NULL.

Esempio

L'esempio seguente mostra la sezione di un file di configurazione di rete (`qm.ini`) che specifica i dettagli richiesti per il Native HA CRR per una delle tre istanze del gruppo live:

```
NativeHALocalInstance:
  ...
  GroupName=NHACRR_live
  GroupRole=live

NativeHARecoveryGroup:
  GroupName=NHACRR_recovery
  ReplicationAddress=recovery-cluster.fyre.ibm.com(4445)
```

Informazioni correlate

NativeHALocalInstance stanza del file qm.ini

NativeHARecoveryGroup stanza del file qm.ini

NativeHAInstance stanza del file qm.ini

Distribuzione e configurazione dei gestori code nei contenitori

È possibile eseguire una serie di attività per distribuire e configurare i gestori code IBM MQ .

Informazioni su questa attività

Per iniziare a distribuire e configurare i gestori code, consultare i seguenti argomenti.

Procedura

- “Distribuzione e configurazione dei gestori code utilizzando IBM MQ Operator” a pagina 97
- “Distribuzione e configurazione dei gestori di code utilizzando il diagramma Helm di esempio” a pagina 187

OpenShift Distribuzione e configurazione dei gestori code utilizzando IBM MQ Operator

Esempi di configurazione; configurazione di HA; connessione dall'esterno di un cluster OpenShift ; integrazione con il pannello di controllo CP4i ; integrazione con la traccia Instana; creazione di un'immagine con file MQSC e INI personalizzati; aggiunta di annotazioni ed etichette personalizzate.

Informazioni su questa attività

Procedura

- “Esempi per configurare un gestore code” a pagina 100.
- “Configurazione dell'alta disponibilità per i gestori code utilizzando IBM MQ Operator” a pagina 111.
- “Configurazione di un instradamento per la connessione a un gestore code dall'esterno di un cluster Red Hat OpenShift” a pagina 148.
- “Integrazione di IBM MQ con la traccia IBM Instana” a pagina 151.
- “Creazione di un'immagine con file MQSC e INI personalizzati, utilizzando la CLI Red Hat OpenShift” a pagina 159.
- “Aggiunta di annotazioni ed etichette personalizzate alle risorse del gestore code” a pagina 163.
- “Disabilitazione dei controlli webhook di runtime” a pagina 163.
- “Disabilitazione degli aggiornamenti dei valori predefiniti per la specifica del gestore code” a pagina 164.

OpenShift CP4I Distribuzione di un gestore code semplice utilizzando IBM MQ Operator

Questo esempio distribuisce un gestore code "quick start", che utilizza la memoria effimera (non persistente) e disattiva la sicurezza IBM MQ . I messaggi non vengono resi persistenti durante i riavvii del gestore code. È possibile modificare la configurazione per modificare molte impostazioni del gestore code.

Informazioni su questa attività

Questa attività offre 3 opzioni per la distribuzione di un gestore code su OpenShift:

1. Distribuire un gestore code con la console OpenShift.
2. Distribuire un gestore code con la CLI OpenShift.
3. Distribuire il gestore code con IBM Cloud Pak for Integration Platform UI.

Procedura

- **Opzione 2: distribuire un gestore code con la console OpenShift .**

  a) Distribuire un gestore code.

  a. Accedi alla console OpenShift con le tue credenziali di amministratore del cluster Red Hat OpenShift Container Platform .
  b. Modifica **Project** nello spazio dei nomi in cui hai installato IBM MQ Operator. Selezionare lo spazio dei nomi dall'elenco a discesa **Progetto** .
  c. Nel riquadro di navigazione, fare clic su **Operatori** > **Operatori installati**.
  d. Nell'elenco del pannello Operatori installati, individuare e fare clic su **IBM MQ**.
  e. Fare clic sulla scheda **Gestore code** .
  f. Fare clic su **Crea QueueManager** . Viene visualizzato il pannello di creazione dell'istanza che offre due metodi per configurare la risorsa: la **vista Modulo** e **vista YAML**. La **Vista modulo** è selezionata per impostazione predefinita.

  b) Configurare il gestore code.

  Passo 2 Opzione 1: configurare nella vista **Modulo**.

  La **Vista modulo** apre un modulo che è possibile utilizzare per visualizzare o modificare la configurazione della risorsa.

  a. Accanto a **Licenza**, fare clic sulla freccia per espandere la sezione di accettazione della licenza.
  b. Impostare **Accetta licenza** su **true** se si accetta l'accordo di licenza.
  c. Fare clic sulla freccia per aprire l'elenco a discesa e selezionare una licenza. IBM MQ è disponibile con diverse licenze. Per ulteriori informazioni sulle licenze valide, consultare “mq.ibm.com/v1beta1: Versioni di licenza attuali” a pagina 248. È necessario accettare la licenza per distribuire un gestore code.
  d. 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` .

  Passo 2 Opzione 2: configurare nella vista **YAML**.

  La **vista YAML** apre un editor contenente un file YAML di esempio per un `QueueManager`. Aggiornare i valori nel file seguendo la procedura riportata di seguito.

  a. Modificare `metadata.namespace` nel nome del proprio progetto (spazio dei nomi).
  b. Modificare il valore di `spec.license.license` nella stringa di licenza che corrisponde ai propri requisiti. Consultare “mq.ibm.com/v1beta1: Versioni di licenza attuali” a pagina 248 per i dettagli della licenza.
  c. Modificare `spec.license.accept` in `true` se si accetta l'accordo di licenza.

d. 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` .

c) Verificare la creazione del gestore code.

È possibile verificare di aver creato un gestore code completando la seguente procedura:

a. Assicurati di essere nello spazio dei nomi in cui hai creato il tuo IBM MQ Operator .

b. Dalla schermata **Home** , fare clic su **Operatori** > **Operatori installati**, quindi selezionare il IBM MQ Operator installato per cui è stato creato il gestore code.

c. Fare clic sulla scheda **Gestore code** . La creazione è completa quando lo stato di `QueueManager` è `Running`.

- **Opzione 2: distribuire un gestore code con la CLI OpenShift .**

a) Crea un file YAML di `QueueManager`

Ad esempio, per installare il gestore code di base in IBM Cloud Pak for Integration, creare il file "mq-quickstart.yaml" con i seguenti contenuti:

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: quickstart-cp4i
spec:
  version: 9.4.3.0-r1
  license:
    accept: false
    license: L-CYPF-CRPF3H
    use: NonProduction
  web:
    enabled: true
  queueManager:
    name: "QUICKSTART"
    storage:
      queueManager:
        type: ephemeral
```

Importante: Se si accetta l'accordo di licenza, modificare `accept: false` in `accept: true`. Consultare “mq.ibm.com/v1beta1: Versioni di licenza attuali” a pagina 248 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 in IBM Cloud Pak for Integration. Per il funzionamento di Single Sign - On, è necessario installare prima altri componenti IBM Cloud Pak for Integration . Consultare “Installazione di IBM MQ Operator per l'utilizzo con CP4I” a pagina 65.

Per installare un gestore code di base indipendentemente da IBM Cloud Pak for Integration, creare il file "mq-quickstart.yaml" con il seguente contenuto:

Nota: Questo esempio utilizza una licenza IBM MQ Advanced , ma è possibile utilizzare anche una licenza IBM MQ . Per ulteriori informazioni, consultare “Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator” a pagina 160.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: quickstart
spec:
  version: 9.4.3.0-r1
  license:
    accept: false
    license: L-NUUP-23NH8Y
  web:
    enabled: true
  queueManager:
    name: "QUICKSTART"
    storage:
      queueManager:
        type: ephemeral
```

Importante: se si accetta l'accordo di licenza di MQ , modificare `accept: false` in `accept: true`. Consultare “mq.ibm.com/v1beta1: Versioni di licenza attuali” a pagina 248 per i dettagli sulla licenza.

b) Creare l'oggetto `QueueManager` .

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

c) Verificare la creazione del gestore code.

Verificare di aver creato un gestore code completando la seguente procedura:

a. Convalidare la distribuzione:

```
oc describe queuemanager Queue_Manager_Resource_Name
```

b. Controllare lo stato:

```
oc describe queuemanager quickstart
```

- **Opzione 3: distribuire un gestore code con IBM Cloud Pak for Integration Platform UI.**

  Segui le istruzioni in Distribuzione di un'istanza utilizzando l'IU della piattaforma.

Attività correlate

“Configurazione di un instradamento per la connessione a un gestore code dall'esterno di un cluster Red Hat OpenShift” a pagina 148
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 217
Come connettersi al IBM MQ Console di un gestore code distribuito su un cluster Red Hat OpenShift Container Platform .

“Esempi per configurare un gestore code” a pagina 100
È possibile configurare un gestore code modificando il contenuto della risorsa personalizzata QueueManager .

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 100
- “Esempio: configurazione di un gestore code con autenticazione TLS reciproca” a pagina 106

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 si sta distribuendo il gestore delle code.

Nota: Un segreto Kubernetes deve essere utilizzato quando il file MQSC o INI contiene dati sensibili.

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 queue manager con la seguente configurazione, utilizzando la riga di comando o la console web Red Hat OpenShift Container Platform .

Nota: Questo esempio utilizza una licenza IBM MQ Advanced , anche se è possibile utilizzare una licenza IBM MQ . Per ulteriori informazioni, consultare “Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator” a pagina 160.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: mqsc-ini-qm
spec:
  version: 9.4.3.0-r1
  license:
    accept: false
    license: L-NUUP-23NH8Y
    use: Production
  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 MQ Advanced , modificare `accept: false` in `accept: true`. Vedere “mq.ibm.com/v1beta1: Versioni di licenza attuali” a pagina 248 per i dettagli sulla licenza.

Informazioni aggiuntive:

- Un gestore code può essere configurato in modo da utilizzare una singola Kubernetes ConfigMap o un segreto (come mostrato in questo esempio) o più 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 l' `example1.mqsc` e viene sempre elaborato prima dell' `example2.mqsc` e, indipendentemente dall'ordine in cui compaiono nella configurazione del gestore della coda.
- 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.
- Quando un pod del gestore code è in esecuzione, le modifiche a Kubernetes ConfigMap non vengono prese in considerazione perché IBM MQ Operator non è a conoscenza della modifica. Se si apportano modifiche alla ConfigMap, ad esempio ai comandi MQSC o ai file INI, è necessario riavviare manualmente i gestori di code per recepire tali modifiche. Per i gestori code a istanza singola, eliminare il pod per attivare il riavvio richiesto. Per le distribuzioni della HA nativa, riavvia prima i pod standby eliminandoli. Quando sono di nuovo in uno stato di esecuzione, elimina il pod attivo per riavviarlo. Questo ordine di riavvii garantisce un tempo di inattività minimo per il gestore code.

OpenShift CP4I *Esempio: fornitura di file aggiuntivi a un gestore di code*

Questo esempio crea un file Kubernetes ConfigMap che contiene due file. Viene quindi distribuito un gestore di code che monta questi file nel contenitore dell' IBM MQ e nelle posizioni specificate.

Informazioni su questa attività

Quando viene utilizzato un gestore di code, è possibile fornire file aggiuntivi. Questi file devono essere definiti in uno o più file Kubernetes ConfigMaps e Secrets. Questi devono essere creati nello spazio dei nomi (progetto) in cui si sta distribuendo il gestore delle code. Kubernetes . Quando i file contengono dati sensibili, è necessario utilizzare un protocollo di sicurezza.

Nota: Questa funzione è supportata solo con l' IBM MQ Operator e 3.5.0 o versioni successive.

Esempio

L'esempio seguente crea un file Kubernetes ConfigMap che contiene due file. Viene quindi distribuito un gestore di code che monta questi file nel contenitore dell' IBM MQ e nelle posizioni specificate.

Esempio ConfigMap: Applica il seguente YAML nel tuo cluster.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: files-example
data:
  file1.txt: |
    Test File 1
  file2.txt: |
    Test File 2
```

Esempio QueueManager: Distribuisci il tuo queue manager con la seguente configurazione, utilizzando la riga di comando o la console web Red Hat OpenShift Container Platform .

Nota: Questo esempio utilizza una licenza IBM MQ Advanced , ma è possibile utilizzare anche una licenza IBM MQ . Per ulteriori informazioni, consultare “Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator” a pagina 160.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: files-qm
spec:
  version: 9.4.3.0-r1
  license:
    accept: false
    license: L-NUUP-23NH8Y
    use: Production
  queueManager:
    files:
    - configMap:
        name: files-example
        defaultMountPath: "/etc/mqm/myfiles"
        items:
```

```
        - item: file1.txt
        - item: file2.txt
          mountPath: "/etc/mqm/otherfiles"
    storage:
      queueManager:
        type: ephemeral
```

Importante: Se si accetta il contratto di licenza dell' IBM MQ Advanced , cambiare `accept: false` in `accept: true`. Vedere “mq.ibm.com/v1beta1: Versioni di licenza attuali” a pagina 248 per i dettagli sulla licenza.

Posizione dei file:

- Il file `file1.txt` è montato nella posizione `/etc/mqm/myfiles/file1.txt`, che è l' **defaultMountPath** a specificata per questo ConfigMap.
- Il file `file2.txt` è montato nella posizione `/etc/mqm/otherfiles/file2.txt`, che è l'override di **mountPath** per questo specifico file.

Informazioni aggiuntive:

- Quando **defaultMountPath** è impostato su (per un ConfigMap o Secret) e l'override **mountPath** non è specificato, i file vengono montati per default nella posizione `/etc/mqm/customfiles`
- Se si modifica il contenuto di uno qualsiasi dei file in " ConfigMap " o "Secret", è necessario riavviare manualmente i gestori di coda per acquisire tali modifiche. Per i gestori di coda a istanza singola, eliminare il pod per attivare il riavvio richiesto. Per le distribuzioni Native HA, riavviare prima i pod in stand-by eliminandoli. Quando sono di nuovo in stato di funzionamento, cancellare il pod attivo per riavviarlo. Questo ordine di riavvio garantisce tempi di inattività minimi per il responsabile della coda.

OpenShift CP4I *Esempio: fornitura di variabili ambientali aggiuntive per un gestore di code*

Questo esempio utilizza un gestore di code con due variabili di ambiente aggiuntive. Queste variabili ambientali sono impostate sul contenitore dell' IBM MQ .

Informazioni su questa attività

Quando viene utilizzato un gestore di code, è possibile fornire ulteriori variabili ambientali. Ciò consente l'impostazione di variabili ambientali che non sono già esposte dall' IBM MQ Operator.

Nota: Questa funzione è supportata solo con l' IBM MQ Operator e 3.5.0 o versioni successive.

Esempio

L'esempio seguente utilizza un gestore di code con due variabili d'ambiente aggiuntive. Queste variabili ambientali sono impostate sul contenitore dell' IBM MQ .

Esempio QueueManager: Distribuisci il tuo queue manager con la seguente configurazione, utilizzando la riga di comando o la console web Red Hat OpenShift Container Platform .

Nota: Questo esempio utilizza una licenza IBM MQ Advanced , ma è possibile utilizzare anche una licenza IBM MQ . Per ulteriori informazioni, consultare “Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator” a pagina 160.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: env-qm
spec:
  version: 9.4.3.0-r1
  license:
    accept: false
    license: L-NUUP-23NH8Y
    use: Production
  queueManager:
    env:
    - name: TEST_ENV_1
      value: "true"
```

```
    - name: TEST_ENV_2
      value: "false"
  storage:
    queueManager:
      type: ephemeral
```

Importante: Se si accetta il contratto di licenza dell' IBM MQ Advanced , cambiare `accept: false` in `accept: true`. Vedere “mq.ibm.com/v1beta1: Versioni di licenza attuali” a pagina 248 per i dettagli sulla licenza.

Informazioni aggiuntive:

- Se si specifica una variabile di ambiente che altrimenti sarebbe stata impostata dall' IBM MQ Operator, ciò potrebbe portare a un comportamento imprevedibile. In questo scenario, viene aggiunta una condizione di stato di avviso al tuo QueueManager.

OpenShift CP4I *Creazione di una PKI autofirmata utilizzando OpenSSL*

IBM MQ ti permette di utilizzare il TLS reciproco per l'autenticazione, dove entrambe le estremità di una connessione forniscono un certificato e i dettagli nel certificato vengono utilizzati per stabilire un'identità con il gestore code. Questo argomento illustra come creare un PKI (Public Key Infrastructure) di esempio utilizzando lo strumento della riga comandi OpenSSL , creando due certificati che possono essere utilizzati in altri esempi.

Prima di iniziare

Verificare che lo strumento della riga comandi OpenSSL sia installato.

Installa IBM MQ cliente aggiungi `samp/bin` e `bin` al *PATH*. È necessario il comando '**runmqicred**, che può essere installato 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/mq94redistclients
- macOS Per Mac: scaricare e configurare IBM MQ MacOS Toolkit: https://developer.ibm.com/tutorials/mq-macos-dev/

Informazioni su questa attività

Importante: Gli esempi descritti di seguito non sono adatti per un ambiente di produzione e sono intesi solo come esempi per andare avanti rapidamente. La gestione dei certificati è un argomento complesso per gli utenti avanzati. Per la produzione, devi considerare cose come la rotazione, la revoca, la lunghezza della chiave, il ripristino di emergenza e molto altro ancora.

Questi passi sono stati verificati utilizzando OpenSSL 3.1.4.

Procedura

1. Creare una chiave privata da utilizzare per la propria autorità di certificazione interna

   ```
   openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:4096 -out ca.key
   ```

   Una chiave privata per l'autorità di certificazione interna viene creata in un file denominato *ca.key*. Questo file deve essere mantenuto sicuro e segreto - verrà utilizzato per firmare i certificati per la propria autorità di certificazione interna.
2. Emettere un certificato autofirmato per l'autorità di certificazione interna

   ```
   openssl req -x509 -new -nodes -key ca.key -sha512 -days 30 -subj "/CN=example-selfsigned-ca"
   -out ca.crt
   ```

   `-days` specifica il numero di giorni di validità del certificato CA root.

   Un certificato viene creato in un file denominato *ca.crt*. Questo certificato contiene le informazioni pubbliche sull'autorità di certificazione interna ed è liberamente condivisibile.

3. Crea una chiave privata e un certificato per un gestore code
   a) Crea una chiave privata e una richiesta di firma certificato per un gestore code

```
openssl req -new -nodes -out example-qm.csr -newkey rsa:4096 -keyout example-qm.key -subj
'/CN=example-qm'
```

   Una chiave privata viene creata in un file denominato *example-qm.key*e una richiesta di firma del certificato viene creata in un file denominato *example-qm.csr*

   b) Firma la chiave gestore code con la tua autorità di certificazione interna

```
openssl x509 -req -in example-qm.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out
example-qm.crt -days 7 -sha512
```

   `-days` specifica il numero di giorni di validità del certificato.

   Un certificato firmato viene creato in un file denominato *example-qm.crt*

   c) Crea un segreto Kubernetes con il certificato e la chiave del gestore code

```
kubectl create secret generic example-qm-tls --type="kubernetes.io/tls" --from-
file=tls.key=example-qm.key --from-file=tls.crt=example-qm.crt --from-file=ca.crt
```

   Viene creato un Kubernetes segreto denominato *example - qm - tls* . Questo segreto contiene la chiave privata per il gestore code, il certificato pubblico e il certificato CA.

4. Crea una chiave privata e un certificato per una applicazione
   a) Crea una chiave privata e una richiesta di firma certificato per un'applicazione

```
openssl req -new -nodes -out example-app1.csr -newkey rsa:4096 -keyout example-app1.key
-subj '/CN=example-app1'
```

   Una chiave privata viene creata in un file denominato *example-app1.key*e una richiesta di firma del certificato viene creata in un file denominato *example-app1.csr*

   b) Firma la chiave gestore code con la tua autorità di certificazione interna

```
openssl x509 -req -in example-app1.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out
example-app1.crt -days 7 -sha512
```

   `-days` specifica il numero di giorni di validità del certificato.

   Un certificato firmato viene creato in un file denominato *example-app1.crt*

   c) Crea una memoria chiavi PKCS#12 con la chiave e il certificato dell'applicazione

   IBM MQ utilizza un database di chiavi e non singoli file di chiavi. Il gestore code inserito nel contenitore creerà il database delle chiavi per il gestore code da un segreto, ma per le applicazioni client è necessario creare manualmente il database delle chiavi.

```
openssl pkcs12 -export -in "example-app1.crt" -name "example-app1" -certfile "ca.crt"
-inkey "example-app1.key" -out "example-app1.p12" -passout pass:PASSWORD
```

   Dove *PASSWORD* è una password di propria scelta.

   Viene creato un keystore in un file denominato *example-app1.p12*. La chiave e il certificato dell'applicazione sono memorizzati all'interno, con una "etichetta" o un "nome descrittivo" di "example-app1", così come il certificato CA.

   d) Se stai utilizzando un arm64 Apple Mac, devi configurare un file aggiuntivo che combina l'applicazione e i certificati CA.
   Ad esempio:

```
cat example-app1.crt ca.crt > example-app1-chain.crt
```

Attività correlate

“Esempio: configurazione di un gestore code con autenticazione TLS reciproca” a pagina 106

Questo esempio distribuisce un gestore code in OpenShift Container Platform utilizzando IBM MQ Operator. Il TLS reciproco viene utilizzato per l'autenticazione, per eseguire l'associazione da un certificato TLS a un'identità nel gestore code.

“Esempio: configurazione della HA nativa utilizzando IBM MQ Operator” a pagina 112
Questo esempio distribuisce un gestore code utilizzando la funzione di alta disponibilità nativa in OpenShift Container Platform utilizzando IBM MQ Operator. Il TLS reciproco viene utilizzato per l'autenticazione, per eseguire l'associazione da un certificato TLS a un'identità nel gestore code.

“Configurazione di un gestore code a più istanze utilizzando IBM MQ Operator” a pagina 145
Questo esempio distribuisce un gestore code a più istanze utilizzando OpenShift Container Platform utilizzando IBM MQ Operator. Il TLS reciproco viene utilizzato per l'autenticazione, per eseguire l'associazione da un certificato TLS a un'identità nel gestore code.

Esempio: configurazione di un gestore code con autenticazione TLS reciproca

Questo esempio distribuisce un gestore code in OpenShift Container Platform utilizzando IBM MQ Operator. Il TLS reciproco viene utilizzato per l'autenticazione, per eseguire l'associazione da un certificato TLS a un'identità nel gestore code.

Prima di iniziare

Per completare questo esempio, è necessario prima aver completato i prerequisiti riportati di seguito:

- Creare un progetto / spazio dei nomi 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 OpenShift Container Platform. Descrive inoltre i passi aggiuntivi richiesti per distribuire il gestore code con TLS abilitato.

Procedura

1. Creare una coppia di certificati come descritto in “Creazione di una PKI autofirmata utilizzando OpenSSL” a pagina 104.
2. Crea una mappa di configurazione contenente comandi MQSC e un file INI

   Creare una Kubernetes ConfigMap 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.

   Assicurati di essere nello spazio dei nomi che hai creato precedentemente (vedi Prima di iniziare), quindi immetti il seguente YAML nella console web OCP o utilizzando la riga di comando.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: example-tls-configmap
data:
  example-tls.mqsc: |
    DEFINE CHANNEL('MTLS.SVRCONN') CHLTYPE(SVRCONN) SSLCAUTH(REQUIRED)
SSLCIPH('ANY_TLS13_OR_HIGHER') REPLACE
    SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=*') USERSRC(NOACCESS)
ACTION(REPLACE)
    SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=example-app1') USERSRC(MAP)
MCAUSER('app1') ACTION(REPLACE)
    SET AUTHREC PRINCIPAL('app1') OBJTYPE(QMGR) AUTHADD(CONNECT,INQ)
    DEFINE QLOCAL('EXAMPLE.QUEUE') REPLACE
    SET AUTHREC PROFILE('EXAMPLE.QUEUE') PRINCIPAL('app1') OBJTYPE(QUEUE)
AUTHADD(BROWSE,PUT,GET,INQ)
  example-tls.ini: |
    Service:
        Name=AuthorizationService
```

```
        EntryPoints=14
        SecurityPolicy=UserExternal
```

MQSC definisce un canale denominato *MTLS.SVRCONN* e una coda denominata *EXAMPLE.QUEUE*. Il canale è configurato per consentire l'accesso solo ai client che presentano un certificato con un "nome comune" *example-app1*. Questo è il nome comune utilizzato in uno dei certificati creati nel passo "1" a pagina 106. Le connessioni su questo canale con questo nome comune vengono associate a un ID utente *app1*, autorizzato a connettersi al gestore code e ad accedere alla coda di esempio. Il file INI abilita una politica di sicurezza che indica che l'ID utente *app1* non deve necessariamente esistere in un registro utente esterno - esiste solo come nome in questa configurazione.

3. Distribuisci il gestore code

   Creare un nuovo gestore code utilizzando la seguente risorsa personalizzata YAML. Assicurarsi di essere nello spazio dei nomi creato prima di iniziare questa attività, quindi immettere il seguente YAML nella console Web OCP o utilizzando la riga comandi. Verificare che sia stata specificata la licenza corretta e accettarla modificando `false` in `true`.

   Nota: Questo esempio utilizza una licenza IBM MQ Advanced , ma è possibile utilizzare anche una licenza IBM MQ . Per ulteriori informazioni, consultare "Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator" a pagina 160.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: exampleqm
spec:
  license:
    accept: false
    license: L-NUUP-23NH8Y
    use: Production
  queueManager:
    name: EXAMPLEQM
    mqsc:
    - configMap:
        name: example-tls-configmap
        items:
        - example-tls.mqsc
    ini:
    - configMap:
        name: example-tls-configmap
        items:
        - example-tls.ini
    storage:
      queueManager:
        type: ephemeral
  version: 9.4.3.0-r1
  pki:
    keys:
      - name: default
        secret:
          secretName: example-qm-tls
          items:
            - tls.key
            - tls.crt
            - ca.crt
```

   Tenere presente che il segreto *example - qm - tls* è stato creato nel passo "1" a pagina 106e che ConfigMap *example - tls - configmap* è stato creato nel passo "2" a pagina 106

4. 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 exampleqm
```

5. Verifica la connessione al gestore code

   Per confermare che il gestore code è configurato per la comunicazione TLS reciproca, attenersi alla procedura descritta in "Verifica di una connessione TLS reciproca a un gestore code dal tuo laptop" a pagina 108.

Risultati

Congratulazioni, hai correttamente distribuito un gestore code con TLS abilitato e che utilizza i dettagli forniti nel certificato TLS per autenticarsi con il gestore code e fornire un'identità.

Verifica di una connessione TLS reciproca a un gestore code dal tuo laptop

Dopo aver creato un gestore code utilizzando IBM MQ Operator, è possibile verificare che funzioni collegandosi ad esso, inserendo e ricevendo un messaggio. Questa attività illustra come connettersi utilizzando programmi di esempio IBM MQ , eseguendoli su una macchina esterna al cluster Kubernetes , come ad esempio il laptop.

Prima di iniziare

Per completare questo esempio, è necessario aver prima completato i seguenti prerequisiti:

- Installare IBM MQ client. Sono necessari i comandi **amqsputc** e **amqsgetc** , che possono essere installati come parte di IBM MQ client come segue:
  - 

Per Windows e Linux: installare il client ridistribuibile IBM MQ per il proprio sistema operativo da https://ibm.biz/mq94redistclients
  - 

Per Mac: scaricare e configurare IBM MQ MacOS Toolkit: https://developer.ibm.com/tutorials/mq-macos-dev/
- Accertarsi di avere i file di chiave e certificato necessari scaricati in una directory sulla macchina e di conoscere la parola d'ordine del keystore. Ad esempio, questi file vengono creati in “Creazione di una PKI autofirmata utilizzando OpenSSL” a pagina 104:
  - `example-app1.p12`
  - `example-app1-chain.crt` (solo se stai utilizzando un Apple Mac arm64)
- Distribuire un gestore code configurato con TLS al cluster OCP, ad esempio seguendo la procedura in “Esempio: configurazione di un gestore code con autenticazione TLS reciproca” a pagina 106

Informazioni su questa attività

Questo esempio utilizza i programmi di esempio IBM MQ in esecuzione su una macchina esterna al cluster Kubernetes , ad esempio il laptop, per connettersi a QueueManager configurato con TLS e per inserire e richiamare messaggi.

Procedura

1. 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 exampleqm
   ```

2. Trova il nome host del gestore code

   Utilizzare il seguente comando per trovare il nome host completo del gestore code dall'esterno del cluster OCP, utilizzando l'instradamento creato automaticamente: `exampleqm-ibm-mq-qm`:

   ```
   oc get route exampleqm-ibm-mq-qm --template="{{.spec.host}}"
   ```

3. Creare una CCDT (IBM MQ Client Channel Definition Table)

   Creare un file denominato `ccdt.json` con il seguente contenuto:

   ```
   {
       "channel":
       [
   ```

```
        {
            "name": "MTLS.SVRCONN",
            "clientConnection":
            {
                "connection":
                [
                    {
                        "host": "hostname from previous step",
                        "port": 443
                    }
                ],
                "queueManager": "EXAMPLEQM"
            },
            "transmissionSecurity":
            {
              "cipherSpecification": "ANY_TLS13",
              "certificateLabel": "example-app1"
            },
            "type": "clientConnection"
        }
   ]
}
```

La connessione utilizza la porta 443, perché è la porta su cui è in ascolto il router Red Hat OpenShift Container Platform . Il traffico verrà inoltrato al gestore code sulla porta 1414.

Se hai utilizzato un nome di canale diverso, dovrai anche modificarlo. Gli esempi TLS reciproci utilizzano un canale denominato *MTLS.SVRCONN*

Per ulteriori dettagli, consultare Configurazione di una CCDT in formato JSON

4. Creare un file INI client per configurare i dettagli di connessione

   Creare un file denominato `mqclient.ini` nella directory corrente. Questo file verrà letto da **amqsputc** e **amqsgetc**.

```
Channels:
  ChannelDefinitionDirectory=.
  ChannelDefinitionFile=ccdt.json
SSL:
  OutboundSNI=HOSTNAME
  SSLKeyRepository=example-app1.p12
  SSLKeyRepositoryPassword=password you used when creating the p12 file
```

   Assicurarsi di aggiornare la *SSLKeyRepositoryPassword* con la password scelta durante la creazione del file PKCS#12 . Esistono altri modi per impostare la password del keystore, incluso l'utilizzo di una password codificata. Per ulteriori informazioni, consultare Come fornire la password del repository delle chiavi per un IBM MQ MQI client su AIX, Linux, and Windows

   Tenere presente che Red Hat OpenShift Container Platform Router utilizza SNI per instradare le richieste al gestore code IBM MQ . L'attributo *OutboundSNI= HOSTNAME* garantisce che il client IBM MQ includa le informazioni necessarie per il funzionamento del router con l'instradamento predefinito configurato da IBM MQ Operator. 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 148.

5. Se si sta utilizzando un arm64 Apple Mac, è necessario configurare una variabile di ambiente aggiuntiva.

```
export MQSSLTRUSTSTORE=example-app1-chain.crt
```

   Questo file contiene la catena di certificati completa, inclusi i certificati CA e dell'applicazione.

6. Inserire i messaggi nella coda

   Esegui il seguente comando:

```
/opt/mqm/samp/bin/amqsputc EXAMPLE.QUEUE EXAMPLEQM
```

   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** .

7. Richiama i messaggi dalla coda

   Esegui il seguente comando:

   ```
   /opt/mqm/samp/bin/amqsgetc EXAMPLE.QUEUE EXAMPLEQM
   ```

   I messaggi aggiunti nel passo precedente sono stati utilizzati e vengono emessi. Dopo pochi secondi, il comando esce.

Risultati

Congratulazioni, hai verificato correttamente la connessione a un gestore code con TLS abilitato e hai mostrato che puoi inserire e ricevere messaggi in modo sicuro 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 163.

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 MQ Adv. Configurazione dell'alta disponibilità per i gestori code utilizzando IBM MQ Operator

è possibile configurare l'alta disponibilità per i gestori di code utilizzando la soluzione Native HA o la soluzione per gestori di code multistanza.

Informazioni su questa attività

Seguire la procedura per la soluzione di alta disponibilità scelta.

Procedura

- “HA nativa” a pagina 42.
- “Esempio: configurazione della HA nativa utilizzando IBM MQ Operator” a pagina 112.
- “Configurazione di un gestore code a più istanze utilizzando IBM MQ Operator” a pagina 145.

OpenShift MQ Adv. *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-NUUP-23NH8Y
    use: Production
  queueManager:
    availability:
      type: NativeHA
  version: 9.4.3.0-r1
```

Nota: Questo esempio utilizza una licenza IBM MQ Advanced . Da 9.4.2, i gestori di code con licenza IBM MQ possono essere configurati per utilizzare le funzionalità Native HA e Cross-Region Replication aggiungendo licenze supplementari a un gestore di code. Per ulteriori informazioni, vedere "Configurazione dell'HA nativo e della replica interregionale sui gestori di code con licenza IBM MQ utilizzando il software IBM MQ Operator" a pagina 115.

il campo `.spec.queueManager.availability.type` deve essere impostato su `NativeHA`.

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 della HA nativa utilizzando IBM MQ Operator" a pagina 112.

Attività correlate

"Esempio: configurazione della HA nativa utilizzando IBM MQ Operator" a pagina 112
Questo esempio distribuisce un gestore code utilizzando la funzione di alta disponibilità nativa in OpenShift Container Platform utilizzando IBM MQ Operator. Il TLS reciproco viene utilizzato per l'autenticazione, per eseguire l'associazione da un certificato TLS a un'identità nel gestore code.

Esempio: configurazione della HA nativa utilizzando IBM MQ Operator

Questo esempio distribuisce un gestore code utilizzando la funzione di alta disponibilità nativa in OpenShift Container Platform utilizzando IBM MQ Operator. Il TLS reciproco viene utilizzato per l'autenticazione, per eseguire l'associazione da un certificato TLS a un'identità nel gestore code.

Prima di iniziare

Per completare questo esempio, è necessario prima aver completato i prerequisiti riportati di seguito:

- Creare un progetto / spazio dei nomi OpenShift Container Platform (OCP) per questo esempio.
- Sulla riga comandi, accedere al cluster OCP e passare allo spazio dei nomi precedente.
- Assicurarsi che IBM MQ Operator sia installato e sia disponibile nel suddetto spazio dei nomi.

Informazioni su questa attività

Questo esempio fornisce una risorsa personalizzata YAML che definisce un gestore code da distribuire in OpenShift Container Platform. Descrive inoltre i passi aggiuntivi richiesti per distribuire il gestore code con TLS abilitato.

Procedura

1. Creare una coppia di certificati come descritto in "Creazione di una PKI autofirmata utilizzando OpenSSL" a pagina 104.
2. Crea una mappa di configurazione contenente comandi MQSC e un file INI

   Creare una Kubernetes ConfigMap 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.

   Assicurati di essere nello spazio dei nomi che hai creato precedentemente (vedi Prima di iniziare), quindi immetti il seguente YAML nella console web OCP o utilizzando la riga di comando.

   ```
   apiVersion: v1
   kind: ConfigMap
   ```

```
metadata:
  name: example-nativeha-configmap
data:
  example-tls.mqsc: |
    DEFINE CHANNEL('MTLS.SVRCONN') CHLTYPE(SVRCONN) SSLCAUTH(REQUIRED)
SSLCIPH('ANY_TLS13_OR_HIGHER') REPLACE
    SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=*') USERSRC(NOACCESS)
ACTION(REPLACE)
    SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=example-app1') USERSRC(MAP)
MCAUSER('app1') ACTION(REPLACE)
    SET AUTHREC PRINCIPAL('app1') OBJTYPE(QMGR) AUTHADD(CONNECT,INQ)
    DEFINE QLOCAL('EXAMPLE.QUEUE') REPLACE
    SET AUTHREC PROFILE('EXAMPLE.QUEUE') PRINCIPAL('app1') OBJTYPE(QUEUE)
AUTHADD(BROWSE,PUT,GET,INQ)
  example-tls.ini: |
    Service:
        Name=AuthorizationService
        EntryPoints=14
        SecurityPolicy=UserExternal
```

MQSC definisce un canale denominato *MTLS.SVRCONN* e una coda denominata *EXAMPLE.QUEUE*. Il canale è configurato per consentire l'accesso solo ai client che presentano un certificato con un "nome comune" *example-app1*. Questo è il nome comune utilizzato in uno dei certificati creati nel passo “1” a pagina 112. Le connessioni su questo canale con questo nome comune vengono associate a un ID utente *app1*, autorizzato a connettersi al gestore code e ad accedere alla coda di esempio. Il file INI abilita una politica di sicurezza che indica che l'ID utente *app1* non deve necessariamente esistere in un registro utente esterno - esiste solo come nome in questa configurazione.

3. Distribuisci il gestore code

   Creare un nuovo gestore code utilizzando la seguente risorsa personalizzata YAML. Assicurarsi di essere nello spazio dei nomi creato prima di iniziare questa attività, quindi immettere il seguente YAML nella console Web OCP o utilizzando la riga comandi. Verificare che sia stata specificata la licenza corretta e accettarla modificando `false` in `true`.

   Nota:

   Questo esempio utilizza una licenza IBM MQ Advanced . Da IBM MQ 9.4.2, le funzioni Native HA e Cross-Region Replication sono disponibili sui gestori di code che usano una licenza IBM MQ . Per ulteriori informazioni, fare riferimento a “Configurazione dell'HA nativo e della replica interregionale sui gestori di code con licenza IBM MQ utilizzando il software IBM MQ Operator” a pagina 115.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: exampleqm
spec:
  license:
    accept: false
    license: L-NUUP-23NH8Y
    use: Production
  queueManager:
    name: EXAMPLEQM
    availability:
      type: NativeHA
      tls:
        secretName: example-qm-tls
    mqsc:
    - configMap:
        name: example-nativeha-configmap
        items:
        - example-tls.mqsc
    ini:
    - configMap:
        name: example-nativeha-configmap
        items:
        - example-tls.ini
    storage:
      queueManager:
        type: persistent-claim
  version: 9.4.2.0-r2
  pki:
    keys:
      - name: default
        secret:
          secretName: example-qm-tls
```

```
          items:
            - tls.key
            - tls.crt
            - ca.crt
```

Si noti che il segreto *example - qm - tls* è stato creato nel passo “1” a pagina 112e la ConfigMap *example - nativeha - configmap* è stata creata nel passo “2” a pagina 112

Il tipo di disponibilità è impostato su *NativeHA* e lo storage persistente è selezionato. Verrà utilizzata la classe di archiviazione predefinita configurata nel cluster Kubernetes . 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`.

I tre pod in un gestore code HA nativo replicano i dati sulla rete. Questo link non è codificato per impostazione predefinita, ma questo esempio utilizza il certificato del gestore code per codificare il traffico. È possibile specificare un certificato differente per ulteriore sicurezza. Il segreto TLS HA nativo deve essere un segreto TLS Kubernetes , che ha una struttura particolare (ad esempio, la chiave privata deve essere denominata *tls.key*).

4. 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 exampleqm
   ```

5. Verifica la connessione al gestore code

   Per confermare che il gestore code è configurato e disponibile, effettuare le operazioni riportate in “Verifica di una connessione TLS reciproca a un gestore code dal tuo laptop” a pagina 108.

6. Forza l'esito negativo del pod attivo

   Per convalidare il ripristino automatico del gestore code, simula un errore pod:

   a) Visualizza i pod attivi e in standby

      Esegui il seguente comando:

      ```
      oc get pods --selector app.kubernetes.io/instance=exampleqm
      ```

      Nota che, nel campo **READY** , il pod attivo restituisce il valore `1/1`, mentre i pod di replica restituiscono il valore `0/1`.

   b) Elimina il pod attivo

      Esegui il seguente comando, specificando il nome completo del pod attivo:

      ```
      oc delete pod exampleqm-ibm-mq-value
      ```

   c) Visualizza di nuovo lo stato del pod

      Esegui il seguente comando:

      ```
      oc get pods --selector app.kubernetes.io/instance=exampleqm
      ```

   d) Visualizza lo stato del gestore code

      Eseguite il seguente comando, specificando il nome completo del <pod attivo>. Il pod attivo deve essere in stato di pronto, il pod con il ready 1/1 mostrato dal comando al punto c:

      ```
      oc exec -t <active-pod> -- dspmq -o nativeha -x -m EXAMPLEQM
      ```

      Dovresti vedere lo stato che mostra che l'istanza attiva è cambiata, ad esempio:

      ```
      QMNAME(EXAMPLEQM) 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)
      ```

e) Verificare nuovamente la connessione al gestore code

Per confermare che il gestore code è stato ripristinato, effettuare le operazioni riportate in “Verifica di una connessione TLS reciproca a un gestore code dal tuo laptop” a pagina 108.

Risultati

Congratulazioni, hai distribuito correttamente un gestore code con alta disponibilità nativa e autenticazione TLS reciproca e verificato che venga ripristinato automaticamente quando il pod attivo ha esito negativo.

Configurazione di Native HA CRR utilizzando il IBM MQ Operator

La replica interregionale (CRR) nativa HA viene configurata utilizzando l'API (`QueueManager`) e le opzioni avanzate sono disponibili utilizzando un file INI.

Nota: Questo esempio utilizza una licenza IBM MQ Advanced . Da IBM MQ 9.4.2, IBM MQ è possibile configurare i gestori di code con licenza per utilizzare le funzionalità di Native HA CRR aggiungendo licenze aggiuntive a un gestore di code. Per ulteriori informazioni, fare riferimento a “Configurazione dell'HA nativo e della replica interregionale sui gestori di code con licenza IBM MQ utilizzando il software IBM MQ Operator” a pagina 115.

Native HA CRR è configurato utilizzando `.spec.queueManager.availability` nell'API di `QueueManager` . Il campo " `.spec.queueManager.availability.type` " deve essere impostato su " `NativeHA`". Il campo `.spec.queueManager.availability.nativeHAGroups.local.role` è impostato su `Live` o `Recovery` per specificare il ruolo del gruppo locale.

È possibile specificare i dettagli del gruppo remoto nella configurazione nella sezione " `.spec.queueManager.availability.nativeHAGroups.remotes` ". Si forniscono il nome host e la porta del percorso di OpenShift utilizzato dal gruppo remoto in cui i dati verranno replicati. Il gruppo live deve includere almeno la sezione dell' `remotes` , per poter replicare al gruppo di recupero. Il gruppo di recupero deve includere la sezione dell' `remotes` , in modo da poter replicare l'operazione sull'altro sito, se mai dovesse diventare attivo. (Vedi “Commutazione e failover nativi HA CRR” a pagina 235.)

Sotto `.spec.queueManager.availability`, è anche possibile configurare un segreto TLS e cifrature da utilizzare tra le istanze del gestore di code durante la replica. Si tratta di un'operazione fortemente raccomandata, per la quale è disponibile una guida passo passo all'indirizzo “Esempio: configurazione della HA nativa utilizzando IBM MQ Operator” a pagina 112.

È necessario configurare un segreto TLS e cifrature da utilizzare tra i gruppi live e recovery durante la replica. È possibile utilizzare lo stesso segreto specificato per la replica tra istanze oppure è possibile specificare un altro segreto per questo scopo.

Attività correlate

“Esempio: configurazione della HA nativa utilizzando IBM MQ Operator” a pagina 112
Questo esempio distribuisce un gestore code utilizzando la funzione di alta disponibilità nativa in OpenShift Container Platform utilizzando IBM MQ Operator. Il TLS reciproco viene utilizzato per l'autenticazione, per eseguire l'associazione da un certificato TLS a un'identità nel gestore code.

Configurazione dell'HA nativo e della replica interregionale sui gestori di code con licenza IBM MQ utilizzando il software IBM MQ Operator

Da IBM MQ 9.4.2, IBM MQ i gestori di code con licenza possono essere configurati per utilizzare le funzionalità di Native HA Cross-Region Replication (CRR) aggiungendo una licenza supplementare a un gestore di code.

Prima di iniziare

Nota: Queste istruzioni sono valide solo se si utilizza il sito IBM MQ Operator. Se state distribuendo i gestori di code su Kubernetes, consultate invece “Configurazione di Native HA e Cross-Region Replication su gestori di code con licenza IBM MQ in Kubernetes” a pagina 186 .

L'indirizzo IBM License Service è necessario per utilizzare le funzioni CRR nativo HA sui gestori di code con licenza IBM MQ . Vedere il passo Distribuzione di License Service del processo di installazione di IBM MQ Operator .

Importante: Questo compito comporta l'aggiornamento dello YAML del gestore di code, che attiva il riavvio del pod.

Procedura

1. Aggiungere uno o entrambi i CR di `IBMLicensingDefinition` , a seconda che si richieda l'uso in produzione o non in produzione. Questi CR aggiungono ulteriori annotazioni di licenza al gestore di code per le funzionalità del CRR Native HA.

   Ripetere questo passaggio in ogni spazio dei nomi in cui si intende utilizzare la licenza aggiuntiva per le funzionalità Native HA CRR.

   - **Opzione 1:** Aggiungere i CR utilizzando la console OpenShift :
     - Accedere alla console di OpenShift con le credenziali di amministratore del cluster Red Hat OpenShift Container Platform .
     - Cambiate il **Progetto** nello spazio dei nomi in cui è distribuito il vostro gestore di code, selezionando lo spazio dei nomi dall'elenco a discesa **Progetto**.
     - Fare clic sull'icona **Creazione rapida** (segno più) in alto a destra per aprire la pagina **Importa YAML**.
     - Aggiungere il CR `IBMLicensingDefinition` YAML (vedere gli esempi alla fine di questo passo).
     - Fai clic su **Crea**.
     - Ripetere questi passaggi per aggiungere l'altro `IBMLicensingDefinition` CR, se necessario.
   - **Opzione 2:** Aggiungere i CR tramite la riga di comando:
     - Accedere all'interfaccia della riga di comando (CLI) Red Hat OpenShift utilizzando il comando `oc login` .
     - Impostate il vostro `project` (spazio dei nomi) su quello in cui è distribuito il vostro gestore di code:

       ```
       oc project <QUEUE-MANAGER-NAMESPACE>
       ```

     - Creare un file `IBMLicensingDefinition` YAML (vedere gli esempi alla fine di questo passo).
     - Applicare il file `IBMLicensingDefinition` YAML :

       ```
       oc apply -f <IBMLicensingDefinition-UNIQUE-NAME>.yaml
       ```

     - Ripetere questi passaggi per aggiungere l'altro `IBMLicensingDefinition` CR, se necessario.

   Nei seguenti esempi di file `IBMLicensingDefinition` CR YAML, cambiare il valore <QUEUE-MANAGER-NAMESPACE> con il progetto in cui è distribuito il gestore di code.

   - Per uso produttivo:

     ```
     apiVersion: operator.ibm.com/v1
     kind: IBMLicensingDefinition
     metadata:
       name: ibmlicensingdefinition-mq-nha-prod
       namespace: <QUEUE-MANAGER-NAMESPACE>
     spec:
       action: cloneModify
       condition:
     ```

```
    metadata:
      annotations:
        additionalLicense: L-EZBK-Y86SSH
        additionalLicenseUse: Production
  scope: cluster
  set:
    productID: 46a57a4a66ee4b904e9b5544b53a2ba2
    productName: IBM MQ Native HA and Cross-Region Replication Add-on
```

- Per uso non di produzione:

```
apiVersion: operator.ibm.com/v1
kind: IBMLicensingDefinition
metadata:
  name: ibmlicensingdefinition-mq-nha-non-prod
  namespace: <QUEUE-MANAGER-NAMESPACE>
spec:
  action: cloneModify
  condition:
    metadata:
      annotations:
        additionalLicense: L-EZBK-Y86SSH
        additionalLicenseUse: NonProduction
  scope: cluster
  set:
    productID: 525b1331a74d4a209d9f6e9c908e374e
    productName: IBM MQ Native HA and Cross-Region Replication Add-on for Non-Production
Env
```

2. Aggiungete le annotazioni pertinenti al gestore di code `spec.annotations`.

   Importante: L'aggiornamento dello YAML del gestore delle code provoca il riavvio del pod.

   Seguire le istruzioni riportate in “Aggiungere annotazioni ai gestori di code utilizzando l'opzione IBM MQ Operator” a pagina 162 per aggiungere una delle seguenti annotazioni, come richiesto:

   - Per IBM MQ e IBM MQ Diritti di add-on per HA nativo e replica interregionale:

```
spec:
  annotations:
    productID: c661609261d5471fb4ff8970a36bccea
    productName: IBM MQ
    additionalLicense: L-EZBK-Y86SSH
    additionalLicenseUse: Production
```

   - Per IBM MQ for Non-Production Environment e IBM MQ Add-on di replica nativa HA e cross-region per i diritti di non produzione

```
spec:
  annotations:
    productID: 151bec68564a4a47a14e6fa99266deff
    productName: IBM MQ for Non-Production Environment
    additionalLicense: L-EZBK-Y86SSH
    additionalLicenseUse: NonProduction
```

   Ripetere questo passaggio per altri gestori di code, se necessario.

OpenShift MQ Adv. Kubernetes *Esempio: Distribuzione di una semplice configurazione CRR HA nativa con l'operatore IBM MQ*

Questo esempio distribuisce un gestore di code utilizzando una semplice configurazione Native HA Cross-Region Replication (CRR) nel sito OpenShift® Container Platform utilizzando l'operatore IBM MQ .

Prima di iniziare

Per completare questo esempio, è necessario eseguire i seguenti passaggi:

- Alla riga di comando, accedere al cluster OpenShift Container Platform (OCP) in uso.
- Creare due progetti/spazi dei nomi OCP per questo esempio. L'esempio si riferisce ai siti di Londra e Roma. Questi si riferiscono a due spazi dei nomi per Londra e Roma in un unico cluster, oppure a uno spazio dei nomi in ciascun cluster per ogni sito.

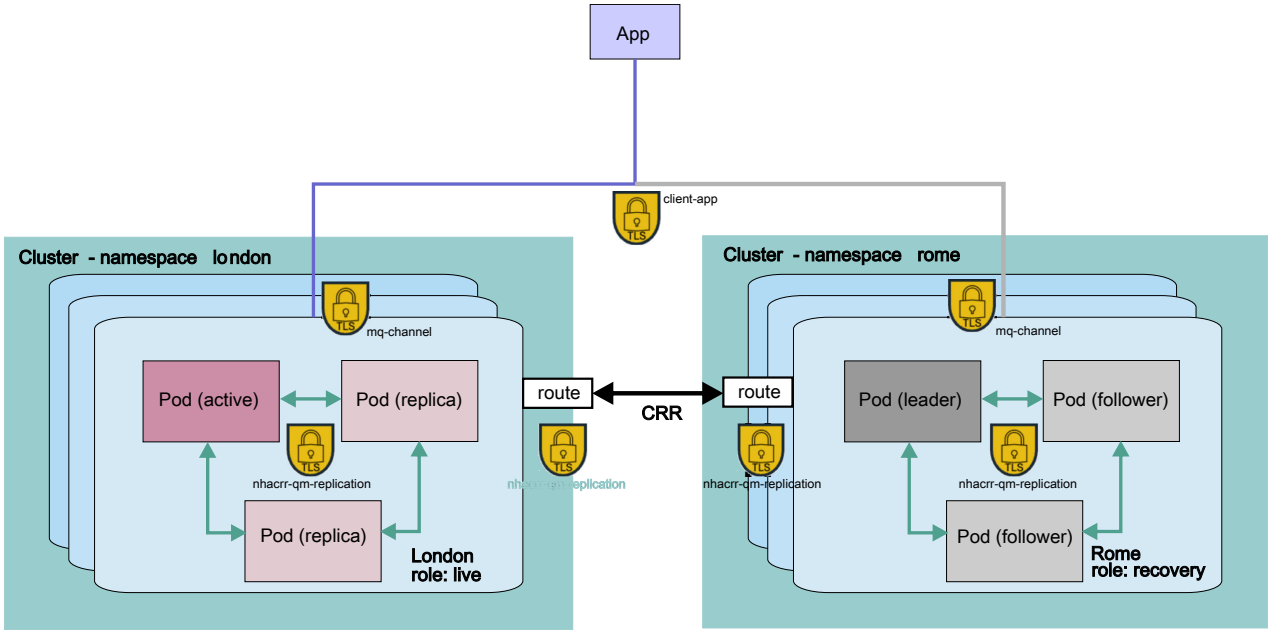
- Assicurarsi che IBM MQ Operator 3.5.0 o superiore sia installato nei cluster e disponibile nei namespace.
- Assicurarsi che sia installato lo strumento da riga di comando OpenSSL.
- Completare il compito “Preparare l'uso di 'Kubernetes creando un segreto di pull” a pagina 86 .
- Leggete “Distribuzione di un gestore code semplice utilizzando IBM MQ Operator” a pagina 98 per acquisire familiarità con l'uso dell'operatore IBM MQ .

Informazioni su questa attività

La configurazione Native HA CRR definisce due gruppi Native HA, che insieme formano un gestore di code altamente disponibile. Questi gruppi sono solitamente distribuiti in due cluster OCP in posizioni diverse, ma questo esempio consente di distribuire entrambi i gruppi nello stesso cluster, se lo si desidera. Per una panoramica della replica interregionale HA nativa, vedere Replicazione interregionale HA nativa.

Per semplicità, questo esempio utilizza un singolo certificato per proteggere tutto il traffico di replica del Queue Manager, denominato nhacrr-qm-replication.

Per capire come vengono utilizzati i certificati in Native HA CRR e come utilizzare certificati diversi per proteggere il traffico di gruppo e di istanza, vedere “Esempio: utilizzo di più certificati TLS con Native HA CRR” a pagina 130.



Importante: Questi esempi servono per iniziare, non sono adatti a un ambiente di produzione. La gestione dei certificati è un argomento complesso, per utenti avanzati. Per la produzione, è necessario considerare aspetti quali la rotazione, la revoca, la lunghezza delle chiavi, il disaster recovery e molto altro ancora.

Questo esempio fornisce la risorsa personalizzata YAML per definire due gruppi HA nativi che forniscono una replica interregionale.

Procedura

1. Creare i certificati da utilizzare per il traffico di replica del gestore delle code.

   Usare i seguenti comandi per creare una chiave privata e usarla per emettere un certificato autofirmato per l'autorità di certificazione interna. Creare quindi una chiave privata e un certificato per il gestore delle code, firmati con l'autorità di certificazione interna:

   ```
   openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:4096 -out nhacrr-ca.key
   openssl req -x509 -new -nodes -key "nhacrr-ca.key" -sha512 -days 30 -subj "/CN=nhacrr-
   selfsigned-ca" -out nhacrr-ca.crt
   openssl req -new -nodes -out nhacrr-qm-replication.csr -newkey rsa:4096 -keyout nhacrr-qm-
   ```

```
replication.key -subj "/CN=nhacrr-qm-replication"
openssl x509 -req -in nhacrr-qm-replication.csr -CA nhacrr-ca.crt -CAkey nhacrr-ca.key
-CAcreateserial -out nhacrr-qm-replication.crt -days 7 -sha512
```

Per ulteriori informazioni su questi comandi, vedere “Creazione di una PKI autofirmata utilizzando OpenSSL” a pagina 104.

2. Creare due segreti Kubernetes contenenti la chiave e i certificati di queue manager. Creare un segreto chiamato nhacrr-qm-replication su entrambi i siti di Roma e Londra, per memorizzare la chiave e il certificato delle istanze di queue manager.

   Utilizzare il seguente comando per creare i segreti. Passare a ogni spazio dei nomi (Roma e Londra) prima di eseguire il comando.

```
kubectl create secret generic nhacrr-qm-replication --type="kubernetes.io/tls" --from-
file=tls.key=nhacrr-qm-replication.key --from-file=tls.crt=nhacrr-qm-replication.crt --from-
file=nhacrr-ca.crt
```

3. Creare una coppia di certificati per il traffico del gestore delle code, come descritto in “Creazione di una PKI autofirmata utilizzando OpenSSL” a pagina 104. Creare il segreto su entrambi i siti di Roma e Londra (si tratta dei certificati denominati mq-channel e client-app nel diagramma).
4. Creare una mappa di configurazione contenente i comandi MQSC e un file INI per Roma e Londra. Creare una risorsa ConfigMap contenente i comandi MQSC per creare una nuova coda di messaggi persistente e un canale SVRCONN e per aggiungere un record di autenticazione del canale che consenta l'accesso al canale. Sul sito di Roma, inserire il seguente YAML nella console web dell'OCP, selezionare ConfigMaps, quindi Creare ConfigMap, o utilizzare il comando **oc apply** . Ripetere l'operazione per il sito di Londra:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: nhacrr-configmap
data:
  nhacrr.mqsc: |
    DEFINE CHANNEL('MTLS.SVRCONN') CHLTYPE(SVRCONN) SSLCAUTH(REQUIRED)
SSLCIPH('ANY_TLS13_OR_HIGHER') REPLACE
    SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=*') USERSRC(NOACCESS)
ACTION(REPLACE)
    SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=example-app1') USERSRC(MAP)
MCAUSER('app1') ACTION(REPLACE)
    SET AUTHREC PRINCIPAL('app1') OBJTYPE(QMGR) AUTHADD(CONNECT,INQ)
    DEFINE QLOCAL('EXAMPLE.QUEUE') DEFPSIST(YES) REPLACE
    SET AUTHREC PROFILE('EXAMPLE.QUEUE') PRINCIPAL('app1') OBJTYPE(QUEUE)
AUTHADD(BROWSE,PUT,GET,INQ)
  nhacrr.ini: |
    Service:
        Name=AuthorizationService
        EntryPoints=14
        SecurityPolicy=UserExternal
```

L'MQSC definisce un canale chiamato MTLS.SVRCONN e una coda chiamata EXAMPLE.QUEUE. Il canale è configurato in modo da consentire l'accesso solo ai client che presentano un certificato con un CN di example-app1, come quello creato al passaggio 3.

5. Distribuire il gruppo Native HA Recovery. Creare un nuovo gruppo di recupero sul sito di Roma. Assicurarsi di essere nello spazio dei nomi roma, quindi inserire il seguente YAML nella console web di OCP per creare QueueManager, o usare il comando **oc apply** . Controllate che sia specificata la licenza corretta e accettate la licenza cambiando `false` in `true`.

   Nota: È necessario utilizzare lo stesso nome di gestore di coda sia nel gruppo live che in quello di recupero. Per impostazione predefinita, se non si specifica `.spec.queuemanager.name`, si sceglie `.metadata.name`. È possibile ottenere nomi corrispondenti utilizzando uno dei seguenti metodi:

   - Nominare la risorsa queue manager con lo stesso nome in ogni cluster (cioè, `.metadata.name` corrisponde) e non sovrascrivere il nome
   - Nominare il gestore di code individualmente in `.metadata.name` e sovrascrivere il nome con `.spec.queuemanager.name` in modo che i nomi corrispondano in entrambi i siti.

In questo esempio, il nome del gestore delle code è impostato su `exampleqm` utilizzando `.metadata.name`.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: exampleqm
  labels:
    role: Recovery
spec:
  license:
    accept: false
    license: L-NUUP-23NH8Y
    use: Production
  queueManager:
    name: EXAMPLEQM
    availability:
      type: NativeHA
      tls:
        secretName: nhacrr-qm-replication
      nativeHAGroups:
        local:
          name: "rome"
          role: "Recovery"
    mqsc:
    - configMap:
        name: nhacrr-configmap
        items:
        - nhacrr.mqsc
    ini:
    - configMap:
        name: nhacrr-configmap
        items:
        - nhacrr.ini
    storage:
      queueManager:
        type: persistent-claim
  version:  9.4.2.0-r1
  pki:
    keys:
      - name: default
        secret:
          secretName: example-qm-tls
          items:
            - tls.key
            - tls.crt
            - ca.crt
```

6. Verificare che il gruppo di recupero sia stato distribuito correttamente:
   a) Emettere il comando `oc get qmgr exampleqm` per visualizzare la risorsa queue manager e attendere che il queue manager sia in esecuzione. Dovrebbe essere visualizzato il seguente messaggio:

   ```
   NAME            PHASE
   exampleqm   Running
   ```

   b) Quando il gestore di code è in funzione, verificare gli stati dei pod utilizzando il comando `oc get pods` . Una capsula deve essere in stato di pronto:

   ```
   NAME                 READY   STATUS    RESTARTS   AGE
   exampleqm-ibm-mq-0   1/1     Running   0          2m45s
   exampleqm-ibm-mq-1   0/1     Running   0          2m45s
   exampleqm-ibm-mq-2   0/1     Running   0          2m45s
   ```

   c) Controllare lo stato del gruppo di recupero utilizzando il comando `oc exec -t <leader-pod> -- dspmq -o nativeha -g` dove `<leader-pod>` è il nome del pod 1/1 pronto (exampleqm-ibm-mq-0 nell'esempio).

   ```
   QMNAME(EXAMPLEQM)                                         ROLE(Leader)
   INSTANCE(exampleqm-ibm-mq-0) INSYNC(yes) QUORUM(1/3) GRPLSN() GRPNAME(rome)
   GRPROLE(Recovery)
    GRPNAME(rome) GRPROLE(Recovery) GRPADDR(Unknown) GRPVER(9.4.2.0) GRSTATUS(Waiting
   for connection) RCOVLSN() RCOVTIME() BACKLOG(Unknown) INSYNC(Unknown) SYNCTIME()
   ALTDATE(2025-04-25) ALTTIME(13.48.15)
   ```

Le parti fondamentali di questo stato sono:

- `Role(Leader)`
- `QUORUM(1/3)` - questo rimane 1/3 fino a quando i gruppi Live e Recovery non si collegano
- `GRSTATUS(Waiting for connection)` - si aggiorna quando il gruppo Live si connette

7. Distribuire il gruppo Native HA Live. Il gruppo Live è dislocato nella sede di Londra:

   Per creare il nuovo gruppo live sul sito di Londra, assicurarsi di essere nello spazio dei nomi corretto, quindi inserire il seguente YAML nella console web di OCP o utilizzando la riga di comando. Verificare che sia stata specificata la licenza corretta e accettarla cambiando false in true.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: exampleqm
  labels:
    role: Live
spec:
  license:
    accept: false
    license: L-NUUP-23NH8Y
    use: Production
  queueManager:
    name: EXAMPLEQM
    availability:
      type: NativeHA
      tls:
        secretName: nhacrr-qm-replication
      nativeHAGroups:
        local:
          name: "london"
          role: "Live"
        remotes:
          - name: "rome"
            addresses:
              - "<rome group hostname>:443"
    mqsc:
    - configMap:
        name: nhacrr-configmap
        items:
        - nhacrr.mqsc
    ini:
    - configMap:
        name: nhacrr-configmap
        items:
        - nhacrr.ini
    storage:
      queueManager:
        type: persistent-claim
  version: 9.4.2.0-r1
  pki:
    keys:
      - name: default
        secret:
          secretName: example-qm-tls
          items:
            - tls.key
            - tls.crt
            - ca.crt
```

   Aggiornare l'indirizzo nello YAML che specifica l'indirizzo che il gruppo live usa per connettersi al gruppo di recupero ("<gruppo chrome hostname>:443 " nell'esempio). Sostituirlo con l'attributo host della rotta exampleqm-ibm-mq-nhacrr sul sito di Roma. Per recuperare questo attributo, assicurarsi di essere sul sito di Roma e utilizzare il seguente comando per recuperare il nome host della rotta nhacrr:

```
oc get route exampleqm-ibm-mq-nhacrr --template="{{.spec.host}}"
```

   Nota:

   Quando vengono distribuiti i gruppi CRR vengono create due rotte, una che termina con `qm` e una che termina con `nhacrr`. La rotta `qm` è quella utilizzata per inviare i messaggi al gestore delle code,

mentre la rotta `nhacrr` è utilizzata per la comunicazione tra gruppi. Aggiungere sempre gli indirizzi host della rotta `nhacrr` come indirizzo remoto del gruppo nello YAML.

8. Verificare il gruppo Live e la Cross Region Replication. Ripetere le operazioni del punto 6, ma questa volta per il gruppo di Londra.
   a) Emettere il comando `oc get qmgr exampleqm` per visualizzare la risorsa queue manager e attendere che il queue manager sia in esecuzione. Dovrebbe essere visualizzato il seguente messaggio:

   ```
   NAME            PHASE
   exampleqm   Running
   ```

   b) Quando il gestore di code è in funzione, verificare gli stati dei pod utilizzando il comando `oc get pods` . Una capsula deve essere in stato di pronto:

   ```
   NAME                 READY   STATUS    RESTARTS   AGE
   exampleqm-ibm-mq-0   1/1     Running   0          2m45s
   exampleqm-ibm-mq-1   0/1     Running   0          2m45s
   exampleqm-ibm-mq-2   0/1     Running   0          2m45s
   ```

   c) Controllare lo stato del gruppo Live utilizzando il comando `oc exec -t <active-pod> -- dspmq -o nativeha -g` dove `<active-pod>` è il nome del pod 1/1 pronto (exampleqm-ibm-mq-0 nell'esempio).

   ```
   QMNAME(EXAMPLEQM)                                         ROLE(Active)
   INSTANCE(exampleqm-ibm-mq-0) INSYNC(yes) QUORUM(3/3) GRPLSN(<0:0:87:58322>)
   GRPNAME(london) GRPROLE(Live)
    GRPNAME(london) GRPROLE(Live) GRPADDR(Unknown) GRPVER(9.4.2.0) GRSTATUS(Normal)
   RCOVLSN(<0:0:87:58322>) RCOVTIME(2025-05-16T14:27:09.051799Z) INITLSN(<0:0:9:4696>)
   INITTIME(2025-05-12T16:58:13.582274Z) LIVETIME(2025-05-12T16:58:16.319844Z)
   ALTDATE(2025-05-16) ALTTIME(14.27.10)
    GRPNAME(rome) GRPROLE(Recovery) GRPADDR(exampleqm-ibm-mq-nhacrr-
   <namespace>.apps.<cluster>) GRPVER(9.4.2.0) CONNGRP(yes) GRSTATUS(Normal)
   RCOVLSN(<0:0:87:58322>) RCOVTIME(2025-05-16T14:27:09.051799Z) BACKLOG(0) INSYNC(yes)
   SYNCTIME(2025-05-16T14:28:38.084565Z) ALTDATE(2025-05-16) ALTTIME(14.28.31)
   ```

   Le parti fondamentali di questo stato sono:

   - `Role(Active)`
   - `QUORUM(3/3)` - il gruppo live ha il quorum pieno
   - `GRSTATUS(Normal)`

   d) Verificare nuovamente lo stato del gruppo di recupero eseguendo il comando `oc exec -t <leader-pod> -- dspmq -o nativeha -g` sul sito di Roma.

   ```
   QMNAME(EXAMPLEQM)                                         ROLE(Leader)
   INSTANCE(exampleqm-ibm-mq-0) INSYNC(yes) QUORUM(3/3) GRPLSN(<0:0:87:58322>)
   GRPNAME(rome) GRPROLE(Recovery)
    GRPNAME(rome) GRPROLE(Recovery) GRPADDR(exampleqm-ibm-mq-nhacrr-
   <namespace>.apps.<cluster>) GRPVER(9.4.3.0) GRSTATUS(Normal)
   RCOVLSN(<0:0:87:58322>) RCOVTIME(2025-05-16T14:27:09.051799Z) BACKLOG(0) INSYNC(yes)
   SYNCTIME(2025-05-16T14:30:40.134627Z) ALTDATE(2025-05-16) ALTTIME(14.27.17)
    GRPNAME(london) GRPROLE(Live) GRPADDR(Unknown) GRPVER(9.4.3.0) CONNGRP(yes)
   GRSTATUS(Normal) RCOVLSN(<0:0:87:58322>) RCOVTIME(2025-05-16T14:27:09.051799Z)
   INITLSN(<0:0:9:4696>) INITTIME(2025-05-12T16:58:13.582274Z)
   LIVETIME(2025-05-12T16:58:16.319844Z) ALTDATE(2025-05-16) ALTTIME(14.30.41)
   ```

   Si noti che i campi `QUORUM(3/3)` e `GRSTATUS(Normal)` sono stati aggiornati.

9. Testare la connessione al gestore di code. Per confermare che il gestore di code è configurato e disponibile, seguire i passaggi in "Verifica di una connessione TLS reciproca a un gestore code dal tuo laptop" a pagina 108, assicurarsi che il nome dell'host del gestore di code, determinato nel passaggio 2, sia il nome dell'host della rotta nel sito di Londra.
10. Aggiungere un indirizzo remoto al gruppo Native HA Recovery. Quando Roma diventa il gruppo live, deve conoscere l'indirizzo remoto di Londra per potersi connettere a Londra come gruppo di recupero. Per aggiungere questo, aggiornare il sito `spec.queueManager.availability.nativeHAGroups` per Roma, con la seguente aggiunta, sotto la stanza locale:

```
          remotes:
            - name: "london"
              addresses:
              - "<london group hostname>:443"
```

Sostituire l'indirizzo con l'attributo host della rotta `exampleqm-ibm-mq-nhacrr` sul sito di Londra. Assicurarsi di essere sul sito di Londra e utilizzare il seguente comando per recuperare il nome host della rotta nhacrr:

```
oc get route exampleqm-ibm-mq-nhacrr --template="{{.spec.host}}"
```

Dopo aver effettuato l'aggiornamento, attendere che i pod si riavviino, siano in funzione e che un pod sia il pod 1/1 pronto.

11. Aggiungere il nome host del sito di Roma al file `ccdt.json` . Dopo uno switchover, in cui Roma diventa il sito live, i clienti devono connettersi al gruppo di Roma. L'aggiunta dell'hostname di Roma al file `ccdt.json` facilita la riconnessione.

    Assicurarsi di essere sul sito di Roma e utilizzare il seguente comando per recuperare il nome host della rotta `qm` :

    ```
    oc get route exampleqm-ibm-mq-qm --template="{{.spec.host}}"
    ```

    Nota: Esistono due percorsi, in questo caso è il percorso per il traffico dei messaggi che deve essere utilizzato. È necessario aggiungere gli indirizzi host della rotta `qm` al file `ccdt.json` .

    Nel file `ccdt.json` , aggiungete una seconda coppia di hostname e porte all'elenco delle connessioni, in modo che l'elenco delle connessioni abbia ora il seguente aspetto:

    ```
    "connection":
                    [
                        {
                            "host": "<hostname from London qm route>",
                            "port": 443
                        },
                        {
                            "host": "<hostname from Rome qm route>",
                            "port": 443
                        }
                    ],
    ```

12. Aggiungere alcuni messaggi persistenti alla coda. La coda definita nel file `mqsc` e specificata nel file ConfigMap ha l'attributo `DEFPSIST(YES)` , quindi tutti i messaggi inviati alla coda saranno persistenti e replicati al gruppo di recupero. Inserite i messaggi nella coda utilizzando il seguente comando:

    ```
    /opt/mqm/samp/bin/amqsputc EXAMPLE.QUEUE EXAMPLEQM
    ```

    Se la connessione al gestore di code ha successo, viene emessa la seguente risposta:

    ```
    target queue is EXAMPLE.QUEUE
    ```

    Inserite diversi messaggi nella coda, immettendo del testo e premendo ogni volta **Invio**. Per terminare, premere due volte **Invio**.

13. Eseguire uno switchover gestito dal gruppo Live a Londra al gruppo Recovery a Roma per dimostrare che la configurazione Native HA - CRR è corretta. Seguire le istruzioni in “Completare una commutazione pianificata su una configurazione Native HA CRR” a pagina 235. Quando si aggiornano i ruoli dei gruppi Live e Recovery, potrebbe essere utile aggiornare `metadata.labels.role` nello YAML per riflettere la modifica, ma questo non è necessario per il passaggio. Dopo aver completato il passaggio, è possibile recuperare i messaggi dal nuovo gruppo live, a Roma.

    Per ulteriori informazioni su failover e switchover, vedere “Commutazione e failover nativi HA CRR” a pagina 235.

14. Recuperare i messaggi inviati prima della commutazione eseguendo il seguente comando:

```
/opt/mqm/samp/bin/amqsgetc EXAMPLE.QUEUE EXAMPLEQM
```

I messaggi aggiunti al punto 11 sono disponibili e vengono emessi. Dopo alcuni secondi, il comando esce.

Operazioni successive

Una volta completata la manutenzione, è possibile tornare al gruppo live originale ripetendo la procedura ai punti 12 e 13.

Si può anche tentare un failover non pianificato utilizzando le istruzioni riportate in “Completare un failover non pianificato su una configurazione Native HA CRR” a pagina 236.

Per saperne di più su come utilizzare diversi certificati per proteggere le diverse parti della configurazione del CRR, vedere “Esempio: utilizzo di più certificati TLS con Native HA CRR” a pagina 130.

Esempio: Aggiunta di un gruppo di recupero a una configurazione Native HA esistente tramite l'operatore IBM MQ

Questo esempio distribuisce un gestore di code con una semplice configurazione Native HA nel sito OpenShift® Container Platform, utilizzando l'operatore IBM MQ . L'esempio aggiunge un gruppo di recupero e poi aggiorna il Native HA queue manager come gruppo live, estendendo la configurazione a una Native HA Cross-Region Replication (CRR).

Prima di iniziare

Per completare questo esempio, è necessario eseguire i seguenti passaggi:

- Alla riga di comando, accedere al cluster OpenShift Container Platform (OCP) in uso.
- Creare due progetti/spazi dei nomi OCP per questo esempio. L'esempio si riferisce ai siti di Londra e Roma. Questi si riferiscono a due spazi dei nomi per Londra e Roma in un unico cluster, oppure a uno spazio dei nomi in ciascun cluster per ogni sito.
- Assicurarsi che IBM MQ Operator 3.5.0 o superiore sia installato nei cluster e disponibile nei namespace.
- Assicurarsi che sia installato lo strumento da riga di comando OpenSSL.
- Completare il compito “Preparare l'uso di 'Kubernetes creando un segreto di pull” a pagina 86 .
- Leggete “Distribuzione di un gestore code semplice utilizzando IBM MQ Operator” a pagina 98 per acquisire familiarità con l'uso dell'operatore IBM MQ .

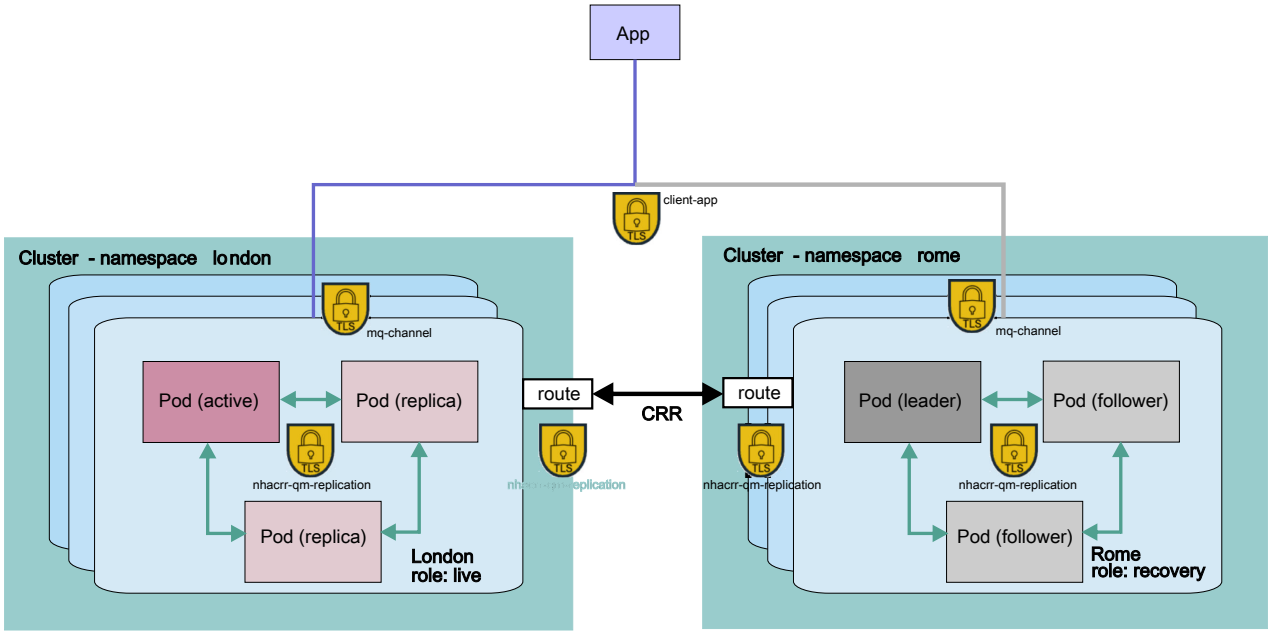
Informazioni su questa attività

I gruppi CRR sono solitamente distribuiti in due cluster OCP in posizioni diverse, tuttavia questo esempio consente di distribuire entrambi i gruppi nello stesso cluster, se lo si desidera. Per una panoramica della replica interregionale HA nativa, vedere Replicazione interregionale HA nativa.

Per semplicità, questo esempio utilizza un singolo certificato per proteggere tutto il traffico di replica del Queue Manager, denominato nhacrr-qm-replication.

Per capire come vengono utilizzati i certificati in Native HA CRR e per utilizzare certificati diversi per proteggere il traffico dei gruppi e delle istanze, consultare “Esempio: utilizzo di più certificati TLS con Native HA CRR” a pagina 130 Deploying Native HA CRR with multiple certificates.

Importante: Questi esempi servono per iniziare, non sono adatti a un ambiente di produzione. La gestione dei certificati è un argomento complesso, per utenti avanzati. Per la produzione, è necessario considerare aspetti quali la rotazione, la revoca, la lunghezza delle chiavi, il disaster recovery e molto altro ancora.



Questo esempio fornisce la risorsa personalizzata YAML per definire due gestori di coda e gruppi di recupero Native HA utilizzati nella replica interregionale.

Procedura

1. Creare un gestore di code Native HA sul sito di Londra. Assicurarsi di essere nello spazio dei nomi di Londra sul cluster corretto e creare una configurazione Native HA seguendo le istruzioni riportate in “Esempio: configurazione della HA nativa utilizzando IBM MQ Operator” a pagina 112. (Nel completare questo passaggio, si creano i due certificati denominati client-app e mq-channel nel diagramma)
2. Creare i certificati da utilizzare per il traffico di replica del gestore di code nella configurazione del CRR. Usare i seguenti comandi per creare una chiave privata e usarla per emettere un certificato autofirmato per l'autorità di certificazione interna. Creare quindi una chiave privata e un certificato per il gestore delle code, firmati con l'autorità di certificazione interna.

   ```
   openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:4096 -out nhacrr-ca.key
   openssl req -x509 -new -nodes -key "nhacrr-ca.key" -sha512 -days 30 -subj "/CN=nhacrr-
   selfsigned-ca" -out nhacrr-ca.crt
   openssl req -new -nodes -out nhacrr-qm-replication.csr -newkey rsa:4096 -keyout nhacrr-qm-
   replication.key -subj "/CN=nhacrr-qm-replication"
   openssl x509 -req -in nhacrr-qm-replication.csr -CA nhacrr-ca.crt -CAkey nhacrr-ca.key
   -CAcreateserial -out nhacrr-qm-replication.crt -days 7 -sha512
   ```

   Per ulteriori informazioni su questi comandi, vedere “Creazione di una PKI autofirmata utilizzando OpenSSL” a pagina 104.
3. Creare due segreti Kubernetes contenenti la chiave e i certificati di queue manager. Creare un segreto chiamato nhacrr-qm-replication su entrambi i siti di Roma e Londra, per memorizzare la chiave e il certificato delle istanze di queue manager. Utilizzate il comando seguente per creare i segreti, passando a ogni spazio dei nomi (Roma, eccetto Londra) prima di eseguire il comando.

   ```
   kubectl create secret generic nhacrr-qm-replication --type="kubernetes.io/tls" --from-
   file=tls.key=nhacrr-qm-replication.key --from-file=tls.crt=nhacrr-qm-replication.crt --from-
   file=nhacrr-ca.crt
   ```
4. Aggiornare Native HA Queue Manager per utilizzare il nuovo segreto per il traffico replicato. Per aggiornare il certificato utilizzato per il traffico interno al gruppo e successivamente per il traffico da gruppo a gruppo, assicurarsi di essere nello spazio dei nomi di Londra, quindi modificare l'attributo `.spec.queuemanager.availability.tls.secretname` nello YAML di QueueManager nella console web di OCP in:

```
      tls:
        secretName: nhacrr-qm-replication
```

5. Creare un segreto nel sito di Roma, contenente il certificato e la chiave generati in precedenza nel passo 1 di "Esempio: configurazione della HA nativa utilizzando IBM MQ Operator" a pagina 112. Creare il segreto accedendo al cluster, passare al namespace Rome ed eseguire il seguente comando nella directory in cui è stato creato il file del certificato:

```
   kubectl create secret generic example-qm-tls --type="kubernetes.io/tls" --from-
file=tls.key=example-qm.key --from-file=tls.crt=example-qm.crt --from-file=ca.crt
```

6. Creare una mappa di configurazione contenente i comandi MQSC e un file INI in Rome. Duplicare la mappa di configurazione creata sul sito di Londra a Roma. Assicurarsi di essere nello spazio dei nomi di Roma, inserire il seguente YAML nella console web dell'OCP, selezionando `ConfigMaps` e poi `Create ConfigMap`, oppure usando il comando `oc apply` .

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: example-nativeha-configmap
data:
  example-tls.mqsc: |
    DEFINE CHANNEL('MTLS.SVRCONN') CHLTYPE(SVRCONN) SSLCAUTH(REQUIRED)
SSLCIPH('ANY_TLS13_OR_HIGHER') REPLACE
    SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=*') USERSRC(NOACCESS)
ACTION(REPLACE)
    SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=example-app1') USERSRC(MAP)
MCAUSER('app1') ACTION(REPLACE)
    SET AUTHREC PRINCIPAL('app1') OBJTYPE(QMGR) AUTHADD(CONNECT,INQ)
    DEFINE QLOCAL('EXAMPLE.QUEUE') REPLACE
    SET AUTHREC PROFILE('EXAMPLE.QUEUE') PRINCIPAL('app1') OBJTYPE(QUEUE)
AUTHADD(BROWSE,PUT,GET,INQ)
  example-tls.ini: |
    Service:
        Name=AuthorizationService
        EntryPoints=14
        SecurityPolicy=UserExternal
```

   L'MQSC definisce un canale chiamato MTLS.SVRCONN e una coda chiamata EXAMPLE.QUEUE. Il canale è configurato in modo da consentire l'accesso solo ai client che presentano un certificato con un CN di example-app1 come quello memorizzato nell'esempio-qm-tls creato al punto 4.

7. Distribuire il gruppo Native HA Recovery. Creare un nuovo gruppo di recupero sul sito di Roma, utilizzando la seguente risorsa personalizzata YAML. Assicurarsi di essere nello spazio dei nomi Rome, quindi inserire il seguente YAML nella console web di OCP per creare il Queue Manager, oppure utilizzando il comando `oc apply` . Verificare che sia stata specificata la licenza corretta e accettarla cambiando false in true.

   Nota: È necessario utilizzare lo stesso nome di gestore di coda sia nel gruppo live che in quello di recupero. Per impostazione predefinita, se non si specifica `.spec.queuemanager.name`, si sceglie `.metadata.name`. È possibile ottenere nomi corrispondenti utilizzando uno dei seguenti metodi:

   - Assegnare alla risorsa queue manager lo stesso nome in ogni cluster (cioè, `.metadata.name` corrisponde) e non sovrascrivere il nome.
   - Nominare il gestore di code individualmente in `.metadata.name` e sovrascrivere il nome con `.spec.queuemanager.name` in modo che i nomi corrispondano in entrambi i siti.

   In questo esempio, il nome del gestore delle code è impostato su `exampleqm` utilizzando `.metadata.name`.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: exampleqm
  labels:
    role: Recovery
spec:
  license:
    accept: false
```

```
    license: L-NUUP-23NH8Y
    use: Production
  queueManager:
    name: EXAMPLEQM
    availability:
      type: NativeHA
      tls:
        secretName: nhacrr-qm-replication
      nativeHAGroups:
        local:
          name: "rome"
          role: "Recovery"
    mqsc:
    - configMap:
        name: example-nativeha-configmap
        items:
        - example-tls.mqsc
    ini:
    - configMap:
        name: example-nativeha-configmap
        items:
        - example-tls.ini
    storage:
      queueManager:
        type: persistent-claim
  version: 9.4.2.0-r1
  pki:
    keys:
      - name: default
        secret:
          secretName: example-qm-tls
          items:
            - tls.key
            - tls.crt
            - ca.crt
```

8. Verificare che il gruppo sia stato distribuito correttamente (questo comando deve essere eseguito sul sito di Roma).
   a) Emettere il comando `oc get qmgr exampleqm` per visualizzare la risorsa queue manager e attendere che il queue manager sia in esecuzione:

```
NAME            PHASE
exampleqm   Running
```

   b) Quando il gestore di code è in esecuzione, è possibile controllare gli stati dei pod utilizzando `oc get pods`, un pod dovrebbe essere in stato di pronto:

```
NAME                 READY   STATUS    RESTARTS   AGE
exampleqm-ibm-mq-0   1/1     Running   0          2m45s
exampleqm-ibm-mq-1   0/1     Running   0          2m45s
exampleqm-ibm-mq-2   0/1     Running   0          2m45s
```

   c) Per verificare lo stato del gruppo di recupero, eseguire il seguente comando:

```
oc exec -t <leader-pod> -- dspmq -o nativeha -g
```

   Dove `<leader-pod>` è il nome del pod pronto 1/1 (`exampleqm-ibm-mq-0` nell'esempio):

```
QMNAME(EXAMPLEQM)                                         ROLE(Leader)
INSTANCE(exampleqm-ibm-mq-0) INSYNC(yes) QUORUM(1/3) GRPLSN() GRPNAME(rome)
GRPROLE(Recovery)
 GRPNAME(rome) GRPROLE(Recovery) GRPADDR(Unknown) GRPVER(9.4.2.0) GRSTATUS(Waiting
for connection) RCOVLSN() RCOVTIME() BACKLOG(Unknown) INSYNC(Unknown) SYNCTIME()
ALTDATE(2025-04-25) ALTTIME(13.48.15)
```

   Le parti fondamentali di questo stato sono:

   - `Role(Leader)`
   - `QUORUM(1/3)` - questo rimane 1/3 fino a quando i gruppi Live e Recovery non si collegano
   - `GRSTATUS(Waiting for connection)` - si aggiorna quando il gruppo Live si connette

9. Aggiornare Native HA Queue Manager per utilizzare il gruppo Live.
   a) Recuperare l'indirizzo del gruppo di recupero utilizzando il seguente comando sul sito di Roma:

```
oc get route exampleqm-ibm-mq-nhacrr --template="{{.spec.host}}"
```

Nota: Quando vengono distribuiti i gruppi CRR vengono create due rotte, una che termina con `qm` e una che termina con `nhacrr`. La rotta `qm` è quella utilizzata per inviare i messaggi al gestore delle code, mentre quella `nhacrr` è utilizzata per la comunicazione tra gruppi. Assicurarsi di aggiungere gli indirizzi host della rotta `nhacrr` come indirizzo remoto per il gruppo nello YAML sottostante.

b) Sostituire l'indirizzo nello YAML sottostante con il nome dell'host restituito dal comando, che specifica l'indirizzo che il gruppo live utilizza per connettersi al gruppo di recupero. Aggiungere il seguente YAML al sito QueueManager YAML di Londra nella console web OCP, direttamente sotto l'attributo `type: NativeHA` :

```
      nativeHAGroups:
        local:
          name: "london"
          role: "Live"
        remotes:
          - name: "rome"
            addresses:
              - "<rome group hostname>:443"
```

Prima di salvare lo YAML, aggiungere un'etichetta Live role all'inizio delle strofe dei metadati:

```
metadata:
  labels:
    role: Live
```

10. Verificare il gruppo Live e la Cross Region Replication.

a) Sul sito di Londra, eseguire il comando `oc get qmgr exampleqm` per visualizzare la risorsa queue manager e attendere che il queue manager sia in esecuzione. Dovrebbe essere visualizzato il seguente messaggio:

```
NAME            PHASE
exampleqm   Running
```

b) Quando il gestore di code è in funzione, verificare gli stati dei pod utilizzando il comando `oc get pods` . Una capsula deve essere in stato di pronto:

```
NAME                 READY   STATUS    RESTARTS   AGE
exampleqm-ibm-mq-0   1/1     Running   0          2m45s
exampleqm-ibm-mq-1   0/1     Running   0          2m45s
exampleqm-ibm-mq-2   0/1     Running   0          2m45s
```

c) Per verificare lo stato del gruppo live, eseguire il seguente comando:

```
oc exec -t <active-pod> -- dspmq -o nativeha -g
```

Dove <`active-pod`> è il nome del pod pronto 1/1 (`exampleqm-ibm-mq-0` nell'esempio):

```
QMNAME(EXAMPLEQM)                                         ROLE(Active)
INSTANCE(exampleqm-ibm-mq-0) INSYNC(yes) QUORUM(3/3) GRPLSN(<0:0:13:21327>)
GRPNAME(london) GRPROLE(Live)
GRPNAME(london) GRPROLE(Live) GRPADDR(Unknown) GRPVER(9.4.3.0) GRSTATUS(Normal)
RCOVLSN(<0:0:13:21327>) RCOVTIME(2025-06-04T16:22:54.414645Z) INITLSN(<0:0:9:5720>)
INITTIME(2025-06-04T16:12:09.629132Z) LIVETIME(2025-06-04T16:12:11.825065Z)
ALTDATE(2025-06-04) ALTTIME(16.23.01)
GRPNAME(rome) GRPROLE(Recovery) GRPADDR(exampleqm-ibm-mq-nhacrr-
<namespace>.apps.<cluster>) GRPVER(9.4.3.0) CONNGRP(yes) GRSTATUS(Normal)
RCOVLSN(<0:0:13:21327>) RCOVTIME(2025-06-04T16:22:54.414645Z) BACKLOG(0) INSYNC(yes)
SYNCTIME(2025-06-04T16:29:17.206996Z) ALTDATE(2025-06-04) ALTTIME(16.29.12)
```

Le parti fondamentali di questo stato sono:

- `Role(Active)`
- `QUORUM(3/3)` - il gruppo live ha il quorum pieno
- `GRSTATUS(Normal)`

d) Eseguire lo stesso comando sul sito di Roma e osservare che anche `QUORUM(3/3)` e `GRSTATUS(Normal)` sono stati aggiornati:

```
QMNAME(EXAMPLEQM)                                         ROLE(Leader)
INSTANCE(exampleqm-ibm-mq-0) INSYNC(yes) QUORUM(3/3) GRPLSN(<0:0:87:58322>)
GRPNAME(rome) GRPROLE(Recovery)
 GRPNAME(rome) GRPROLE(Recovery) GRPADDR(exampleqm-ibm-mq-nhacrr-
rome.apps.destructive-1.cp.fyre.ibm.com) GRPVER(9.4.3.0) GRSTATUS(Normal)
RCOVLSN(<0:0:87:58322>) RCOVTIME(2025-05-16T14:27:09.051799Z) BACKLOG(0) INSYNC(yes)
SYNCTIME(2025-05-16T14:30:40.134627Z) ALTDATE(2025-05-16) ALTTIME(14.27.17)
 GRPNAME(london) GRPROLE(Live) GRPADDR(Unknown) GRPVER(9.4.3.0) CONNGRP(yes)
GRSTATUS(Normal) RCOVLSN(<0:0:87:58322>) RCOVTIME(2025-05-16T14:27:09.051799Z)
INITLSN(<0:0:9:4696>) INITTIME(2025-05-12T16:58:13.582274Z)
LIVETIME(2025-05-12T16:58:16.319844Z) ALTDATE(2025-05-16) ALTTIME(14.30.41)
```

11. Aggiungere un indirizzo remoto al gruppo Native HA Recovery. Quando Roma diventa il gruppo live, deve conoscere l'indirizzo remoto di Londra per potersi connettere a Londra come gruppo di recupero. Per aggiungerlo, aggiornare il sito `spec.queueManager.availability.nativeHAGroups` per Roma, con quanto segue, sotto la stanza locale:

```
        remotes:
          - name: "london"
            addresses:
              - "<london group hostname>:443"
```

Sostituire <london group hostname> con l'attributo host della rotta `exampleqm-ibm-mq-nhacrr` sul sito di Londra. Assicurarsi di essere sul sito di Londra e utilizzare il seguente comando per recuperare il nome host della rotta nhacrr:

```
oc get route exampleqm-ibm-mq-nhacrr --template="{{.spec.host}}"
```

Dopo aver effettuato l'aggiornamento, attendere che i pod si riavviino, siano in funzione e che un pod sia il pod 1/1 pronto.

12. Aggiungere il nome host del sito di Roma al file `ccdt.json` . Dopo uno switchover, in cui Roma diventa il sito live, i client devono connettersi al gruppo di Roma e l'aggiunta dell'hostname di Roma al file `ccdt.json` consente di farlo.

a) Assicurarsi di essere sul sito di Roma e utilizzare il seguente comando per recuperare il nome host della rotta qm:

oc get route exampleqm-ibm-mq-qm --template="{{.spec.host}}"

Nota: Esistono due percorsi, in questo caso utilizzare il percorso per il traffico di messaggi. Aggiungere gli indirizzi host della rotta `qm` alla rotta `ccdt.json`.

b) Nel file `ccdt.json` , aggiungete una seconda coppia di hostname e porte all'elenco delle connessioni, in modo che l'elenco delle connessioni abbia ora il seguente aspetto:

```
                "connection":
                [
                    {
                        "host": "<hostname from London qm route>",
                        "port": 443
                    },
                    {
                        "host": "<hostname from Rome qm route>",
                        "port": 443
                    }
                ],
```

13. Eseguire uno switchover gestito da Londra a Roma per dimostrare che la configurazione Native HA - CRR è corretta, quindi eseguire uno switchover gestito tra il gruppo Live a Londra e il gruppo Recovery a Roma. Seguire le istruzioni in “Completare una commutazione pianificata su una configurazione Native HA CRR” a pagina 235. Quando si aggiornano i ruoli dei gruppi Live e Recovery, potrebbe essere utile aggiornare `metadata.labels.role` nello YAML per riflettere la modifica, ma questo non è necessario per il passaggio. Dopo aver completato il passaggio, è possibile recuperare i messaggi dal nuovo gruppo live, a Roma.

Per ulteriori informazioni su failover e switchover, vedere “Commutazione e failover nativi HA CRR” a pagina 235.

Operazioni successive

Una volta completata la manutenzione, è possibile tornare al gruppo Live originale ripetendo la procedura ai punti 12 e 13.

Si può anche tentare un failover non pianificato utilizzando le istruzioni riportate in “Completare un failover non pianificato su una configurazione Native HA CRR” a pagina 236.

Infine, per saperne di più su come utilizzare diversi certificati per proteggere le varie parti della configurazione CRR, vedere Configurazione di TLS per Native HA CRR.

Esempio: utilizzo di più certificati TLS con Native HA CRR

Prima di iniziare

Prima di iniziare questo esempio, è necessario completare “Esempio: Distribuzione di una semplice configurazione CRR HA nativa con l'operatore IBM MQ” a pagina 117 (questo esempio presuppone che la distribuzione esista e che il gruppo di Londra sia il gruppo live).

Successivamente, completa la seguente procedura:

- Assicuratevi di poter accedere al cluster o ai cluster OpenShift Container Platform (OCP) che state utilizzando e selezionate il progetto/namespace pertinente. (L'esempio descritto in “Esempio: Distribuzione di una semplice configurazione CRR HA nativa con l'operatore IBM MQ” a pagina 117 ha creato due progetti/namespace OpenShift Container Platform (OCP) in due namespace per Londra e Roma in un unico cluster, oppure un namespace in ciascun cluster per ogni sito. Questo esempio si riferisce anche a questi siti come Londra e Roma)
- Assicurarsi che IBM MQ Operator 3.5.0 o superiore sia installato nei cluster e disponibile nei namespace.
- Assicurarsi che sia installato lo strumento da riga di comando OpenSSL.
- Completare il compito “Preparare l'uso di 'Kubernetes creando un segreto di pull” a pagina 86 .
- Leggete “Distribuzione di un gestore code semplice utilizzando IBM MQ Operator” a pagina 98 per acquisire familiarità con l'uso dell'operatore IBM MQ .

Informazioni su questa attività

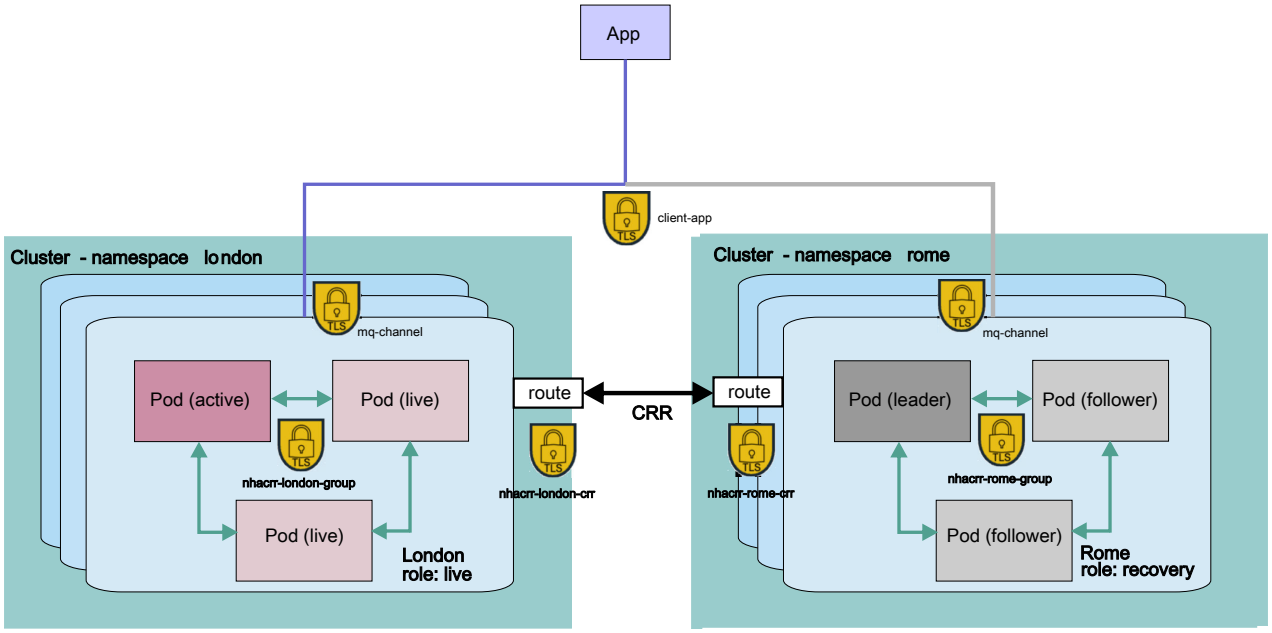
Questo esempio inizia con una panoramica su come vengono utilizzati TLS e i certificati in una configurazione CRR Native HA, quindi utilizza semplici passaggi per dimostrare come viene protetto il traffico di `nhacrr-london-group`, `nhacrr-rome-group`, `nhacrr-london-crr` e `nhacrr-rome-crr` .

La procedura descritta nella sezione seguente illustra come modificare l'implementazione di “Esempio: Distribuzione di una semplice configurazione CRR HA nativa con l'operatore IBM MQ” a pagina 117 per utilizzare certificati diversi per ogni tipo di traffico e controllare quale CipherSpec viene utilizzato.

Configurazione di TLS per CRR HA nativo

Il diagramma seguente mostra dove vengono utilizzati i certificati nella configurazione del CRR Native HA. I certificati vengono utilizzati come segue:

1. Per proteggere la comunicazione tra il gestore di code e le applicazioni client - `client-app` e `mq-channel`
2. Per proteggere il traffico di replica del gestore delle code - `nhacrr-london-group`, `nhacrr-rome-group`, `nhacrr-london-crr` e `nhacrr-rome-crr`



Comunicazione sicura tra il gestore delle code e le applicazioni

I due gruppi HA nativi, mostrati nel diagramma, formano insieme un gestore di code altamente disponibile, al quale le applicazioni client possono connettersi tramite il gruppo live. Quando si verifica una commutazione, le applicazioni vengono scollegate e devono riconnettersi al nuovo gruppo attivo. (L'aggiunta dei dettagli di connessione di entrambi i gruppi a un elenco di connessioni dell'applicazione facilita la riconnessione)

Se le applicazioni che si connettono tramite un percorso sono all'esterno o in un cluster diverso da uno o entrambi i gruppi, devono comunicare tramite TLS utilizzando chiavi private e certificati. Nel diagramma, il certificato e la chiave del client sono contrassegnati da `client-app` e quelli del gestore della coda da `mq-channel`. Il certificato `mq-channel` e la chiave privata devono essere aggiunti come due segreti, uno per ogni gruppo.

“Esempio: Distribuzione di una semplice configurazione CRR HA nativa con l'operatore IBM MQ” a pagina 117 vi guida nella creazione di questi certificati, chiavi e segreti. I passaggi che seguono si concentrano sui certificati per il traffico di replica del gestore di code e non modificano i certificati `mq-channel` o `client-app` .

Protezione del traffico di replica del gestore delle code

Il traffico viene replicato sia all'interno dei gruppi che tra di essi. Il traffico di rete tra i due gruppi deve essere sempre protetto, ma il traffico all'interno di un gruppo può essere in chiaro, poiché il traffico all'interno del cluster non è generalmente accessibile.

1. `nhacrr-london-group` e `nhacrr-rome-group` rappresentano i certificati e le chiavi private utilizzate per proteggere il traffico all'interno di ciascun gruppo.
2. `nhacrr-london-crr` e `nhacrr-rome-crr` rappresentano i certificati e le chiavi private utilizzati per proteggere ciascuna estremità del traffico intergruppo.

Nell' esempio: Distribuzione di una semplice configurazione CRR Native HA utilizzando l'operatore IBM MQ, lo stesso certificato e la stessa chiave privata, memorizzati in un segreto `nhacrr-qm-replication` , in ogni spazio dei nomi, vengono utilizzati per proteggere il traffico di `nhacrr-london-group`, `nhacrr-rome-group`, `nhacrr-london-crr` e `nhacrr-rome-crr` .

Importante: Questi esempi servono per iniziare, non sono adatti a un ambiente di produzione. La gestione dei certificati è un argomento complesso, per utenti avanzati. Per la produzione, è necessario considerare aspetti quali la rotazione, la revoca, la lunghezza delle chiavi, il disaster recovery e molto altro ancora.

Questo esempio mostra come utilizzare certificati diversi per ogni tipo di traffico.

Procedura

1. Creare quattro set di chiavi e certificati da utilizzare per il traffico di replica del gestore di code. Per crearli per questo esempio, eseguire i seguenti comandi. Ogni serie di comandi crea una chiave privata, la utilizza per emettere un certificato autofirmato per un'autorità di certificazione interna, quindi crea una chiave privata e un certificato firmato con l'autorità di certificazione interna. Questo esercizio viene ripetuto per creare i certificati per il traffico di `nhacrr-london-group`, `nhacrr-rome-group`, `nhacrr-london-crr` e `nhacrr-rome-crr` .

   Per una spiegazione più dettagliata dei comandi, vedere "Creazione di una PKI autofirmata utilizzando OpenSSL" a pagina 104.

   Nota: Questo esempio crea CA separate per ogni catena di certificati, per consentire all'esempio di dimostrare l'impostazione della fiducia tra i gruppi, se necessario. I certificati della stessa organizzazione spesso utilizzano una CA comune; in questo caso, le strofe di fiducia nello YAML di Queue Manager non sono necessarie, poiché la CA è già attendibile.

   ```
   openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:4096 -out nhacrr-london-group-
   ca.key openssl req -x509 -new -nodes -key
   "nhacrr-london-group-ca.key" -sha512 -days 30 -subj "/CN=nhacrr-london-group-selfsigned-ca"
   -out nhacrr-london-group-ca.crt openssl
   req -new -nodes -out nhacrr-london-group.csr -newkey rsa:4096 -keyout nhacrr-london-
   group.key -subj "/CN=nhacrr-london-group"
   openssl x509 -req -in nhacrr-london-group.csr -CA nhacrr-london-group-ca.crt -CAkey nhacrr-
   london-group-ca.key -CAcreateserial -out
   nhacrr-london-group.crt -days 7 -sha512
   ```

   ```
   openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:4096 -out nhacrr-rome-group-ca.key
   openssl req -x509 -new -nodes -key
   "nhacrr-rome-group-ca.key" -sha512 -days 30 -subj "/CN=nhacrr-rome-group-selfsigned-ca"
   -out nhacrr-rome-group-ca.crt openssl req -new
   -nodes -out nhacrr-rome-group.csr -newkey rsa:4096 -keyout nhacrr-rome-group.key -subj "/
   CN=nhacrr-rome-group" openssl x509
   -req -in nhacrr-rome-group.csr -CA nhacrr-rome-group-ca.crt -CAkey nhacrr-rome-group-ca.key
   -CAcreateserial -out nhacrr-rome-group.crt
   -days 7 -sha512
   ```

   ```
   openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:4096 -out nhacrr-london-crr-ca.key
   openssl req -x509 -new -nodes -key
   "nhacrr-london-crr-ca.key" -sha512 -days 30 -subj "/CN=nhacrr-london-crr-selfsigned-ca"
   -out nhacrr-london-crr-ca.crt openssl req -new -nodes
   -out nhacrr-london-crr.csr -newkey rsa:4096 -keyout nhacrr-london-crr.key -subj "/CN=nhacrr-
   london-crr" openssl x509 -req -in
   nhacrr-london-crr.csr -CA nhacrr-london-crr-ca.crt -CAkey nhacrr-london-crr-ca.key
   -CAcreateserial -out nhacrr-london-crr.crt -days 7 -sha512
   ```

   ```
   openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:4096 -out nhacrr-rome-crr-ca.key
   openssl req -x509 -new -nodes -key
   "nhacrr-rome-crr-ca.key" -sha512 -days 30 -subj "/CN=nhacrr-rome-crr-selfsigned-ca" -out
   nhacrr-rome-crr-ca.crt openssl req -new -nodes -out
   nhacrr-rome-crr.csr -newkey rsa:4096 -keyout nhacrr-rome-crr.key -subj "/CN=nhacrr-rome-
   crr" openssl x509 -req -in nhacrr-rome-crr.csr
   -CA nhacrr-rome-crr-ca.crt -CAkey nhacrr-rome-crr-ca.key -CAcreateserial -out nhacrr-rome-
   crr.crt -days 7 -sha512
   ```

2. Creare i segreti Kubernetes contenenti le chiavi e i certificati per Roma. Utilizzare i seguenti comandi per creare due segreti TLS sul sito di Roma:

   a. Un certificato per proteggere il traffico interno del gruppo Roma.

   b. Un certificato per proteggere il traffico intergruppo del gruppo Roma.

   ```
   kubectl create secret generic nhacrr-rome-group --type="kubernetes.io/tls" --
   from-file=tls.key=nhacrr-rome-group.key --from-file=tls.crt=nhacrr-rome-group.crt --from-
   file=nhacrr-rome-group-ca.crt
   kubectl create secret generic nhacrr-rome-crr --type="kubernetes.io/tls" --from-
   file=tls.key=nhacrr-rome-crr.key --from-file=tls.crt=nhacrr-rome-crr.crt --from-file=nhacrr-
   rome-crr-ca.crt
   ```

Ora utilizzare i seguenti comandi per creare due segreti contenenti i certificati di Roma per il gruppo di Londra di cui fidarsi:

```
kubectl create secret generic nhacrr-london-group-trust --from-file=tls.crt=nhacrr-london-
group.crt --from-file=ca.crt=nhacrr-london-group-ca.crt --type=opaque
kubectl create secret generic nhacrr-london-crr-trust --from-file=tls.crt=nhacrr-london-
crr.crt  --from-file=ca.crt=nhacrr-london-crr-ca.crt --type=opaque
```

3. Creare Kubernetes secrets contenente le chiavi e i certificati per Londra. Utilizzare i seguenti comandi per creare due segreti TLS sul sito di Londra:

   a. Un certificato per proteggere il traffico interno del gruppo di Londra.

   b. Un certificato per proteggere il traffico intergruppo del gruppo di Londra.

```
kubectl create secret generic nhacrr-london-group --type="kubernetes.io/tls" --from-
file=tls.key=nhacrr-london-group.key --from-file=tls.crt=nhacrr-london-group.crt --from-
file=nhacrr-london-group-ca.crt
kubectl create secret generic nhacrr-london-crr --type="kubernetes.io/tls" --
from-file=tls.key=nhacrr-london-crr.key --from-file=tls.crt=nhacrr-london-crr.crt --from-
file=nhacrr-london-crr-ca.crt
```

   Ora utilizzare i seguenti comandi per creare due segreti contenenti i certificati di Londra per il gruppo di Roma di cui fidarsi:

```
kubectl create secret generic nhacrr-rome-group-trust --from-file=tls.crt=nhacrr-rome-
group.crt --from-file=ca.crt=nhacrr-rome-group-ca.crt --type=opaque
kubectl create secret generic nhacrr-rome-crr-trust --from-file=tls.crt=nhacrr-rome-
crr.crt  --from-file=ca.crt=nhacrr-rome-crr-ca.crt --type=opaque
```

4. Proteggere il traffico nei siti di Roma e Londra con certificati diversi. L'installazione creata in “Esempio: Distribuzione di una semplice configurazione CRR HA nativa con l'operatore IBM MQ” a pagina 117 proteggeva tutto il traffico di replica con il segreto `nhacrr-qm-replication` . Questo passaggio modifica la configurazione per proteggere il traffico all'interno di ciascun gruppo con i segreti `nhacrr-rome-group` e `nhacrr-london-group` rispettivamente.

   L'attributo `.spec.queuemanager.availability.tls.secretName` specifica il segreto da utilizzare per proteggere il traffico all'interno del gruppo del sito. Quando non sono definiti altri attributi TLS, il segreto viene utilizzato anche per proteggere il traffico tra i gruppi.

   a) Sul sito di Roma, modificare l'attributo `.spec.queuemanager.availability.tls.secretName` nello YAML di QueueManager nella console web di OCP in nhacrr-rome-group:

```
      tls:
        secretName: nhacrr-rome-group
```

   b) Aggiungere una stanza `spec.queueManager.availability.NativeHAGroups.remotes.trust` allo YAML e aggiungere il segreto `nhacrr-london-group-trust` come quello di cui fidarsi:

```
...
      nativeHAGroups:
        local:
          name: "rome"
          role: "Recovery"
        remotes:
          - name: "london"
            addresses:
              - "exampleqm-ibm-mq-nhacrr-london.apps.trouble.cp.fyre.ibm.com:443"
            trust:
              - secret:
                  items:
                    - tls.crt
                    - ca.crt
                  secretName: nhacrr-london-group-trust
...
```

   Il segreto `nhacrr-london-group-trust` contiene il certificato e la CA memorizzati nel segreto `nhacrr-london-group` nel sito di Londra. Il segreto `nhacrr-london-group` viene utilizzato

per proteggere il traffico dal gruppo di Londra, quindi il gruppo di Roma utilizza questo segreto per confermare l'affidabilità del certificato fornito.

c) Completare i passaggi equivalenti sul sito di Londra. Modificare l'attributo `.spec.queuemanager.availability.tls.secretName` in:

```
      tls:
        secretName: nhacrr-london-group
```

d) Aggiungere il segreto `nhacrr-rome-group-trust` a una stanza `spec.queueManager.availability.NativeHAGroups.remotes.trust` :

```
...
      nativeHAGroups:
        local:
          name: "london"
          role: "Live"
        remotes:
          - name: "rome"
            addresses:
              - "<rome group hostname>:443"
            trust:
              - secret:
                  items:
                    - tls.crt
                    - ca.crt
                  secretName: nhacrr-rome-group-trust
...
```

5. Verificare la Cross Region Replication con i nuovi certificati. Nel sito di Londra, emettere il comando `oc get qmgr exampleqm` per visualizzare la risorsa queue manager e attendere che il queue manager sia in esecuzione. Quindi verificare gli stati dei pod utilizzando il comando `oc get pods` . Un pod deve essere in stato di pronto, questo è il pod attivo.

   Successivamente si può lanciare il comando `oc exec -t <active-pod> -- dspmq -o nativeha -g` per verificare lo stato del gruppo live. Sostituire `<active-pod>` con il nome del pod 1/1 pronto restituito dal comando `oc get pods` .

```
QMNAME(EXAMPLEQM)                                         ROLE(Active) INSTANCE(exampleqm-
ibm-mq-0) INSYNC(yes) QUORUM(3/3) GRPLSN(<0:0:87:58322>) GRPNAME(london) GRPROLE(Live)
 GRPNAME(london) GRPROLE(Live) GRPADDR(Unknown) GRPVER(9.4.3.0) GRSTATUS(Normal)
RCOVLSN(<0:0:87:58322>) RCOVTIME(2025-05-16T14:27:09.051799Z) INITLSN(<0:0:9:4696>)
INITTIME(2025-05-12T16:58:13.582274Z) LIVETIME(2025-05-12T16:58:16.319844Z)
ALTDATE(2025-05-16) ALTTIME(14.27.10)
 GRPNAME(rome) GRPROLE(Recovery) GRPADDR(exampleqm-ibm-mq-nhacrr-
<namespace>.apps.<cluster>) GRPVER(9.4.3.0) CONNGRP(yes) GRSTATUS(Normal)
RCOVLSN(<0:0:87:58322>) RCOVTIME(2025-05-16T14:27:09.051799Z) BACKLOG(0) INSYNC(yes)
SYNCTIME(2025-05-16T14:28:38.084565Z) ALTDATE(2025-05-16) ALTTIME(14.28.31)
```

   Le parti fondamentali di questo stato sono:

   - `Role(Active)`
   - `QUORUM(3/3)` - il gruppo live ha il quorum pieno
   - `GRSTATUS(Normal)`

6. Proteggere il traffico tra i gruppi con certificati separati. È possibile utilizzare certificati separati per il traffico all'interno del gruppo e per quello tra gruppi. In questa fase, i certificati contenuti nei segreti `nhacrr-rome-crr` e `nhacrr-london-crr` vengono utilizzati per proteggere il traffico tra i gruppi, lasciando che i segreti `nhacrr-rome-group` e `nhacrr-london-group` proteggano il traffico all'interno di ciascun gruppo. Per ottenere questo risultato, sono necessarie due modifiche in ogni sito, una per specificare il segreto per il traffico tra i gruppi e una per modificare i trust in modo che corrispondano ai nuovi certificati utilizzati.

   Nota: Se i segreti utilizzati per gli attributi `.spec.queuemanager.availability.NativeHAGroups.local.tls.secretName` e `.spec.queuemanager.availability.tls.secretName` contengono gli stessi certificati, si verifica un conflitto e le istanze di queue manager non sono in grado di avviarsi. Se si desidera utilizzare un singolo certificato, impostare solo `.spec.queuemanager.availability.tls.secretName` .

a) Sul sito di Roma, aggiungere l'attributo`spec.queuemanager.availability.NativeHAGroups.local.tls.key.secretName` nello YAML QueueManager nella console web OCP e impostarlo su `nhacrr-rome-crr`. Quindi cambiare `.spec.queueManager.availability.NativeHAGroups.remotes.trust.secret.secretName` in `nhacrr-london-crr-trust`:

```
...
      nativeHAGroups:
        local:
          name: "rome"
          role: "Recovery"
          tls:
            key:
              secretName: nhacrr-rome-crr
        remotes:
          - name: "london"
            addresses:
              - "<london group hostname>:443"
            trust:
              - secret:
                  items:
                    - tls.crt
                    - ca.crt
                  secretName: nhacrr-london-crr-trust
...
```

b) Completare i passaggi equivalenti sul sito di Londra. Aggiungere l'attributo `.spec.queuemanager.availability.NativeHAGroups.local.tls.key.secretName` nello YAML QueueManager nella console web OCP e impostarlo su `nhacrr-london-crr`. Quindi cambiare `.spec.queueManager.availability.NativeHAGroups.remotes.trust.secret.secretName` in `nhacrr-rome-crr-trust`.

```
...
      nativeHAGroups:
        local:
          name: "london"
          role: "Live"
          tls:
            key:
              secretName: nhacrr-london-crr
        remotes:
          - name: "london"
            addresses:
              - "<london group hostname>:443"
            trust:
              - secret:
                  items:
                    - tls.crt
                    - ca.crt
                  secretName: nhacrr-rome-crr-trust
...
```

7. Verificare la Cross Region Replication con i nuovi certificati. Nel sito di Londra, eseguire il comando `oc get qmgr exampleqm` per visualizzare la risorsa queue manager e attendere che il queue manager sia in esecuzione. Quindi controllare gli stati dei pod utilizzando `oc get pods`, anche in questo caso un pod dovrebbe essere in stato di pronto, questo è il pod attivo.

   Eseguire il comando `oc exec -t <active-pod> -- dspmq -o nativeha -g` per verificare lo stato del gruppo live. Sostituire `<active-pod>` con il nome del pod pronto 1/1 restituito dal comando `oc get pods`.

```
QMNAME(EXAMPLEQM)                                         ROLE(Active) INSTANCE(exampleqm-
ibm-mq-0) INSYNC(yes) QUORUM(3/3) GRPLSN(<0:0:87:58322>) GRPNAME(london) GRPROLE(Live)
 GRPNAME(london) GRPROLE(Live) GRPADDR(Unknown) GRPVER(9.4.3.0) GRSTATUS(Normal)
RCOVLSN(<0:0:87:58322>) RCOVTIME(2025-05-16T14:27:09.051799Z) INITLSN(<0:0:9:4696>)
INITTIME(2025-05-12T16:58:13.582274Z) LIVETIME(2025-05-12T16:58:16.319844Z)
ALTDATE(2025-05-16) ALTTIME(14.27.10)
 GRPNAME(rome) GRPROLE(Recovery) GRPADDR(exampleqm-ibm-mq-nhacrr-
<namespace>.apps.<cluster>) GRPVER(9.4.3.0) CONNGRP(yes) GRSTATUS(Normal)
```

```
RCOVLSN(<0:0:87:58322>) RCOVTIME(2025-05-16T14:27:09.051799Z) BACKLOG(0) INSYNC(yes)
SYNCTIME(2025-05-16T14:28:38.084565Z) ALTDATE(2025-05-16) ALTTIME(14.28.31)
```

Le parti fondamentali di questo stato sono:

- `Role(Active)`
- `QUORUM(3/3)` - il gruppo live ha il quorum pieno
- `GRSTATUS(Normal)`

Infine, se si lancia il comando `kubectl logs <active-pod>` , si dovrebbero vedere i seguenti messaggi nel log, che confermano l'uso dei certificati corretti per i diversi tipi di traffico. Per il traffico intergruppo, si dovrebbe vedere:

```
AMQ3307I: The Native HA secure connection between the local group and 'rome' is using
certificate DN 'CN=nhacrr-london-crr'
```

E per il traffico del gruppo interno, si dovrebbero vedere i seguenti messaggi, dove `<replica-pod>` è il nome di uno dei pod con lo stato di ready 0/1:

```
AMQ3306I: The Native HA secure connection between the local instance and '<replica-pod>'
accepted remote certificate DN 'CN=nhacrr-london-group'
```

8. Aggiungere CipherSpecs specifico per il traffico replicato.

   Se non sono stati definiti CipherSpecs specifici per il traffico replicato, viene utilizzato ANY_TLS13_OR_HIGHER in modo che i protocolli e la crittografia siano negoziati tra le istanze. Per ulteriori informazioni su CipherSpecs che IBM MQ supporta, leggere Abilitazione di CipherSpecs.

   Per specificare un CipherSpec sia per il traffico all'interno dei gruppi che tra i gruppi, si aggiunge l'attributo `.spec.queuemanager.availability.tls.cipherSpec` allo YAML QueueManager.

   Aggiungere il seguente YAML allo YAML QueueManager nella console web dell'OCP in entrambi i siti di Roma e Londra per utilizzare il sito TLS_AES_256_GCM_SHA384 CipherSpec per tutto il traffico di replica:

```
      tls:
        secretName: nhacrr-rome-group
        cipherSpec: "TLS_AES_256_GCM_SHA384"
```

   Se è necessario specificare un CipherSpec separato per il traffico tra i gruppi, l'attributo `.spec.queuemanager.availability.NativeHAGroups.local.tls.cipherSpec` può essere aggiunto allo YAML QueueManager.

   Aggiungere il seguente YAML allo YAML QueueManager nella console web dell'OCP per entrambi i siti di Roma e Londra, in modo da utilizzare TLS_AES_128_GCM_SHA256 CipherSpec per il traffico tra i gruppi e TLS_AES_256_GCM_SHA384 CipherSpec per il traffico all'interno dei gruppi:

```
      nativeHAGroups:
        local:
       ...
          tls:
            key:
              secretName: nhacrr-rome-crr
            cipherSpec: "TLS_AES_128_GCM_SHA256"
```

   Dopo il riavvio delle istanze, eseguire il seguente comando sul sito di Londra:

```
kubectl logs
        <active-pod>
```

   Nel registro dovrebbero apparire messaggi simili a quelli dell'esempio seguente. L'esempio mostra che la comunicazione tra le istanze del gruppo avviene tramite CipherSpec TLS_AES_256_GCM_SHA384:

```
[CommentInsert1(exampleqm-ibm-mq-2), CommentInsert2(exampleqm-ibm-mq-replica-2(9414)),
        CommentInsert3(TLS_AES_256_GCM_SHA384)]
```

Questo esempio mostra che la connessione al gruppo Roma utilizza CipherSpec TLS_AES_128_GCM_SHA256:

```
AMQ3253I: A Native HA recovery group connection to 'rome' has
        been established using the 'TLS_AES_128_GCM_SHA256' CipherSpec.
```

9. Consentire il traffico di testo semplice all'interno di un gruppo. Il traffico tra gruppi deve essere sempre protetto e crittografato, ma il traffico all'interno di un gruppo può essere non crittografato. Impostare `.spec.queuemanager.availability.tls.cipherSpec` su vuoto nel sito di Londra per rimuovere la crittografia per il traffico interno del gruppo di Londra:

   ```
       tls:
         cipherSpec: ''
         secretName: nhacrr-london-group
   ```

   Dopo il riavvio delle istanze, eseguire il seguente comando sul sito di Londra:

   ```
   kubectl logs
           <active-pod>
   ```

   Nel registro viene visualizzato il seguente messaggio:

   `AMQ9722W: Plain text communication is enabled.`

10. Disabilita la sicurezza per il traffico all'interno di un gruppo. La rimozione della stanza `.spec.queuemanager.availability.tls` disabilita completamente la sicurezza per il traffico interno di un gruppo.

    Nota: Il traffico tra i gruppi deve essere sicuro, quindi se `.spec.queuemanager.availability.tls.secretName` viene rimosso è necessario specificare `.spec.queuemanager.availability.NativeHAGroups.local.tls.key.secretName` .

    Disattivare la sicurezza per il gruppo Roma eliminando la stanza `.spec.queuemanager.availability.tls` dallo YAML QueueManager nella console web OCP del sito Roma:

    ```
        tls:
          secretName: nhacrr-rome-group
          cipherSpec: "TLS_AES_256_GCM_SHA384"
    ```

    Dopo il riavvio delle istanze, eseguire il comando `kubectl logs <active-pod>` sul sito di Roma per vedere messaggi simili a questi nel log:

    ```
    AMQ3213I: Native HA inbound connection accepted from 'exampleqm-ibm-mq-2'.
    [CommentInsert1(exampleqm-ibm-mq-2), CommentInsert2(<ip>)]
    AMQ3211I: Native HA outbound connection established to 'exampleqm-ibm-mq-2'.
    [CommentInsert1(exampleqm-ibm-mq-2), CommentInsert2(<replica>(9414))]
    ```

    invece di:

    ```
    AMQ3214I: Native HA inbound secure connection accepted from 'exampleqm-ibm-mq-0'.
    [CommentInsert1(exampleqm-ibm-mq-0), CommentInsert2(<ip>), CommentInsert3(<cipherspec>)]
    AMQ3212I: Native HA outbound secure connection established to 'exampleqm-
    ibm-mq-0'. [CommentInsert1(exampleqm-ibm-mq-0), CommentInsert2(<replica>(9414)),
    CommentInsert3(<cipherspec)
    ```

OpenShift MQ Adv. *Visualizzazione dello stato dei gestori code della HA nativa per i contenitori IBM MQ*

Per i contenitori 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

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, mentre quelle delle istanze di replica potrebbero essere obsolete.

V 9.4.2 Per una configurazione Native HA Cross-Region Replication (CRR), le informazioni fornite dipendono anche dal fatto che l'istanza faccia parte di un gruppo Live o Recovery. Le informazioni del gruppo Live sono più definitive, ma possono essere ancora non aggiornate poiché il gruppo Recovery invia solo informazioni periodiche.

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.
- V 9.4.2 Visualizza lo stato dei gruppi a cui l'istanza appartiene o a cui è collegata.

I seguenti campi di stato sono utilizzati per segnalare lo stato di configurazione di Native HA e Native HA Cross-Region Replication (CRR) di un'istanza di queue manager. Questi campi appaiono sempre:

RUOLO
Specifica il ruolo corrente dell'istanza. È uno di `Active`, `Replica`, `Unknown`, (per un gruppo di recupero) `Leader`, o `Not Configured`.

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*.

V 9.4.2 **GRPLSN**
Specifica il numero di sequenza del log (LSN) che è stato replicato tra un quorum di istanze nel gruppo HA, in formato " *nnnnn:nnnnn:nnnnn:nnnnn* ". Vuoto se l'LSN non è noto.

V 9.4.2 **GRPNAME**
Specifica il nome del gruppo HA. Vuoto se non è stato configurato un nome.

V 9.4.2 **GRPROLE**
Specifica il ruolo attuale del gruppo HA ed è uno dei seguenti:

- `Live` - Un gruppo dal vivo
- `Recovery` - Un gruppo di recupero
- `Pending live` - Un gruppo in attesa di diventare `Live`
- `Pending recovery` - Un gruppo in attesa di diventare `Recovery`
- `Not configured` - L'HA nativo non è stato configurato
- `Unknown` - Il ruolo attuale non può essere determinato

Questi campi appaiono quando si richiedono informazioni estese per tutte le istanze utilizzando il parametro `-x`:

REPLADDR
L'indirizzo di replica dell'istanza del gestore code.

COLLEGA
Indica se l'istanza è connessa all'istanza attiva.

BACKLOG
Indica il numero di KB che l'istanza è dietro l'istanza attiva.

CONNETTIN
Indica se l'istanza denominata è connessa a questa istanza.

ALTDATE
Indica la data dell'ultimo aggiornamento di queste informazioni. Vuoto se non è mai stato aggiornato.

ALTTIME
Indica l'ora dell'ultimo aggiornamento di queste informazioni. Vuoto se non è mai stato aggiornato.

V 9.4.2 **ACKLSN**
Indica l'LSN che l'istanza ha riconosciuto come scritto nel registro di recupero, in formato *nnnnn:nnnnn:nnnnn:nnnnn*. Vuoto se il valore LSN non è noto.

V 9.4.2 **Stato HA**

Indica lo stato operativo di questa istanza ed è uno dei seguenti valori:

- `Normal` - l'istanza funziona normalmente.
- `Checking` - l'istanza viene controllata per garantire la coerenza del registro di ripristino.
- `Synchronizing` - il sistema sta inviando dati per sincronizzarlo.
- `Rebasing` - un nuovo registro di ripristino viene inviato all'istanza.
- `Disk full` - l'istanza ha il disco pieno.
- `Disconnected` - non può comunicare con l'istanza.
- `Unknown` - lo stato dell'istanza è sconosciuto.

V 9.4.2 **SYNCTIME**
Indica l'ora in cui questa istanza è stata sincronizzata l'ultima volta con il gestore della coda attivo, in formato ISO 8601. Vuoto se l'ora non è nota.

V 9.4.2 Questi campi appaiono quando si richiedono informazioni sul gruppo utilizzando il parametro `-g`. Questi campi sono utili per visualizzare lo stato quando il gruppo fa parte di una configurazione Native HA CRR:

V 9.4.2 **GRPNAME**
Specifica il nome del gruppo HA. Vuoto se non è stato configurato un nome.

V 9.4.2 **GRPROLE**
Specifica il ruolo attuale del gruppo HA ed è uno dei seguenti:

- `Live` - Un gruppo dal vivo
- `Recovery` - Un gruppo di recupero
- `Pending live` - Un gruppo in attesa di diventare `Live`
- `Pending recovery` - Un gruppo in attesa di diventare `Recovery`
- `Unknown` - Il ruolo attuale non può essere determinato
- `Not configured` - L'HA nativo non è stato configurato

V 9.4.2 **GRPADDR**
L'indirizzo IP utilizzato per connettersi al gruppo. Viene impostato su `Unknown` se l'indirizzo non è noto perché non si è verificata una connessione al gruppo.

V 9.4.2 **GRPVER**
Indica la versione della capogruppo corrente nel formato `V.R.M.F`, dove `V.R.M.F` è il numero di Version, Release, Modification e Fix Pack. Se il valore non è noto, viene impostato a `?.?.?.?`

V 9.4.2 **CONNGRP**
Indica se il gruppo è collegato al gruppo locale ed è uno dei seguenti:

- `yes` - il gruppo è collegato.
- `no` - il gruppo è disconnesso.
- `unknown` - lo stato della connessione è sconosciuto.
- `suspended` - il gruppo è collegato, ma il gruppo non può replicare i dati finché non viene risolta un'incompatibilità di configurazione tra i gruppi.

Questo campo viene visualizzato solo per i gruppi diversi dal gruppo locale.

V 9.4.2 **INSYNC**

Viene visualizzato solo per un gruppo con il ruolo `Recovery` o `Pending live`. Indica se il gruppo può diventare `Live` senza perdita di dati in caso di failover. (Si noti che un gruppo di `Recovery` potrebbe non essere molto spesso sincronizzato con il gruppo di `Live` perché il registro viene replicato in modo asincrono. Utilizzare i valori RCOVLSN e RCOVTIME per indicare se potrebbe verificarsi una perdita di dati in un failover non pianificato).

V 9.4.2 **SYNCTIME**

Viene visualizzato solo per un gruppo con il ruolo `Recovery` o `Pending live`. Indica l'ora in cui questo gruppo è stato sincronizzato l'ultima volta con il gruppo dell' `Live` , in formato ISO 8601 (vuoto se non è noto). (Si noti che un gruppo di `Recovery` potrebbe non essere molto spesso sincronizzato con il gruppo di `Live` perché il registro viene replicato in modo asincrono. Utilizzare i valori RCOVLSN e RCOVTIME per indicare se potrebbe verificarsi una perdita di dati in un failover non pianificato).

V 9.4.2 **BACKLOG**

Viene visualizzato solo per un gruppo con il ruolo `Recovery` o `Pending live`. Indica il numero di KB che il gruppo è indietro rispetto al gruppo live.

V 9.4.2 **GRSTATO**

Indica lo stato operativo di questo gruppo ed è uno dei seguenti valori:

- `Normal` - il gruppo funziona normalmente.
- `Checking` - il gruppo viene controllato per garantire la coerenza del registro di ripristino.
- `Synchronizing` - il gruppo riceve dati per sincronizzarsi.
- `Rebasing` - un nuovo registro di ripristino viene inviato all'istanza.
- `Waiting for connection` - il gruppo è in attesa di una connessione da un gruppo con il ruolo `Live` o `Pending Recovery`.
- `Partitioned` - due gruppi credono di essere il gruppo Live (cioè uno stato di "cervello diviso").
- `Unknown` - lo stato del gruppo è sconosciuto.

V 9.4.2 **RCOVLSN**

Un numero di sequenza del registro (LSN) che il gruppo potrebbe recuperare, in formato *nnnnn:nnnnn:nnnnn:nnnnn*. Qualsiasi dato scritto nel registro di ripristino dopo questo punto potrebbe andare perso se si verifica un failover non pianificato. Questo valore può essere passato nell'opzione -s del comando **dmpmqlog**.

V 9.4.2 **RCOVTIME**

Un orario a cui il gruppo può recuperare, in formato ISO 8601. Qualsiasi dato scritto nel registro di ripristino dopo questo momento potrebbe andare perso se si verifica un failover non pianificato. Questo valore può essere passato nelle opzioni -t o -u del comando **dmpmqlog**.

V 9.4.2 **INITLSN**

Viene visualizzato solo per un gruppo con il ruolo `Live` o `Pending recovery`. Indica il numero di sequenza del registro (LSN) dell'ultimo record di registro recuperato quando il gruppo Native HA è diventato attivo, nel formato *nnnnn:nnnnn:nnnnn:nnnnn*, o vuoto se il gruppo non è attivo.

V 9.4.2 **Tempo ini**
Viene visualizzato solo per un gruppo con il ruolo `Live` o `Pending recovery`. Indica l'ora dell'ultimo record di registro recuperato quando il gruppo HA nativo è diventato inizialmente attivo, in formato ISO 8601. Vuoto, se il gruppo non è attivo.

V 9.4.2 **TEMPO DI VITA**
Viene visualizzato solo per un gruppo con il ruolo `Live` o `Pending recovery`. Indica l'ora in cui il gruppo è diventato per la prima volta il gruppo live, in formato ISO 8601. Vuoto, se il gruppo non è attivo.

V 9.4.2 **ALTDATE**
Indica la data dell'ultimo aggiornamento di queste informazioni.

V 9.4.2 **ALTTIME**
Indica l'ora dell'ultimo aggiornamento di queste informazioni.

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)
```

V 9.4.2 Un'istanza di BOB che è leader di un gruppo di recupero in una configurazione CRR Native HA riporta il seguente stato:

```
QMNAME(BOB)                STATUS(Recovery group leader)
```

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) V 9.4.2
GRPLSN(<0:25:365:1000>) GRPNAME() GRPROLE(Live)
```

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) V 9.4.2
GRPLSN(<0:25:365:1000>) GRPNAME() GRPROLE(Live)
```

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) V 9.4.2
GRPLSN() GRPNAME() GRPROLE(Live)
```

V 9.4.2 Un'istanza attiva di un gestore di code chiamato BOB appartenente al gruppo Live in una configurazione CRR Native HA potrebbe riportare il seguente stato:

```
QMNAME(BOB)         ROLE(Active) INSTANCE(inst1) INSYNC(yes) QUORUM(3/3)
GRPLSN(<0:25:365:1000>) GRPNAME(alpha) GRPROLE(Live)
```

V 9.4.2 La stessa istanza, quando il gruppo era in attesa di diventare gruppo di recupero, potrebbe riportare il seguente stato:

```
QMNAME(BOB)         ROLE(Active) INSTANCE(inst1) INSYNC(yes) QUORUM(3/3)
GRPLSN(<0:25:365:1000>) GRPNAME(alpha) GRPROLE(Pending recovery)
```

V 9.4.2 Un gestore di code chiamato CAROL che non è un'istanza di una configurazione Native HA o Native HA CRR riporta il seguente stato:

```
QMNAME(CAROL)       ROLE(Not configured) INSTANCE() INSYNC() QUORUM() GRPLSN() GRPNAME()
GRPROLE(Not configured)
```

- 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
```

Ad esempio, se si emette questo comando sul nodo che esegue l'istanza attiva del gestore di code BOB, dove tutte le istanze sono sincronizzate, si potrebbe ricevere il seguente stato:

```
QMNAME(BOB)         ROLE(Active) INSTANCE(inst1) INSYNC(yes) QUORUM(2/3)
GRPLSN(<0:25:365:1000>) GRPROLE(Live) GRPNAME()
 INSTANCE(inst1) ROLE(Active)  REPLADDR(localhost) CONNACTV(yes) INSYNC(yes)
BACKLOG(0)  CONNINST(yes) ACKLSN(<0:26:365:1000>)  HASTATUS(Normal)
SYNCTIME(2023-12-20T17:13:27.853412Z) ALTDATE(2023-12-20) ALTTIME(17.13.27)
 INSTANCE(inst3) ROLE(Replica) REPLADDR(localhost) CONNACTV(yes) INSYNC(yes)
BACKLOG(0)  CONNINST(yes) ACKLSN (<0:24:365:1000>) HASTATUS(Normal)
SYNCTIME(2023-12-20T17:13:27.853412Z) ALTDATE(2023-12-20) ALTTIME(17.13.27)
 INSTANCE(inst2) ROLE(Replica) REPLADDR(localhost) CONNACTV(yes) INSYNC(yes)
BACKLOG(0)  CONNINST(yes) ACKLSN (<0:25:365:1000>) HASTATUS(Synchronizing)
SYNCTIME(2023-12-20T17:13:22.123321Z) ALTDATE(2023-12-20) ALTTIME(17.13.27)
```

In un altro esempio, se si emette questo comando su un nodo che esegue un'istanza di replica del gestore di code BOB, che è in ritardo, si potrebbe ricevere il seguente stato:

```
QMNAME(BOB)         ROLE(Replica) INSTANCE(inst2) INSYNC(no) QUORUM(2/3)
GRPLSN(<0:25:365:1000>) GRPROLE(Live) GRPNAME()
 INSTANCE(inst1) ROLE(Active)  REPLADDR(localhost) CONNACTV(yes) INSYNC(yes)
BACKLOG(0)  CONNINST(yes) ACKLSN(<0:26:365:1000>)  HASTATUS(Normal)
SYNCTIME(2023-12-20T17:13:27.853412Z) ALTDATE(2023-12-20) ALTTIME(17.13.27)
 INSTANCE(inst3) ROLE(Replica) REPLADDR(localhost) CONNACTV(yes) INSYNC(yes)
BACKLOG(0)  CONNINST(yes) ACKLSN (<0:24:365:1000>) HASTATUS(Normal)
SYNCTIME(2023-12-20T17:13:27.853412Z) ALTDATE(2023-12-20) ALTTIME(17.13.27)
 INSTANCE(inst2) ROLE(Replica) REPLADDR(localhost) CONNACTV(yes) INSYNC(no)
BACKLOG(435) CONNINST(yes) ACKLSN (<0:25:365:1000>) HASTATUS(Synchronizing)
SYNCTIME(2023-12-20T17:13:22.123321Z) ALTDATE(2023-12-20) ALTTIME(17.13.27)
```

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) GRPLSN()
GRPROLE() GRPNAME()
```

```
 INSTANCE(inst1) ROLE(Unknown) REPLADDR(9.20.123.45) CONNACTV(Unknown) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ACKLSN()  HASTATUS(Unknown) SYNCTIME() ALTDATE() ALTTIME()
ALTDATE() ALTTIME()
 INSTANCE(inst2) ROLE(Unknown) REPLADDR(9.20.123.46) CONNACTV(Unknown) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ACKLSN()  HASTATUS(Unknown) SYNCTIME() ALTDATE() ALTTIME()
ALTDATE() ALTTIME()
 INSTANCE(inst3) ROLE(Unknown) REPLADDR(9.20.123.47) CONNACTV(No)      INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ACKLSN()  HASTATUS(Unknown) SYNCTIME() ALTDATE() ALTTIME()
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)
```

- V 9.4.2

  Per determinare lo stato operativo di tutti i gruppi Native HA in una configurazione Native HA CRR:

  ```
  oc exec -t Pod dspmq -o nativeha -g -m QMgrName
  ```

  Se si impartisce questo comando a un membro del gruppo Live, denominato alpha, si potrebbe ricevere il seguente stato per alpha e il gruppo Recovery, beta:

  ```
  GRPNAME(alpha) GRPROLE(Live) GRPADDR(9.20.135.129) GRPVER(09040200)
  GRSTATUS(Normal) RCOVLSN(<0:0:104:65163>) RCOVTIME(2023-12-20T17:37:27.114787Z)
  INITLSN(<0:0:101:352>) INITTIME(2023-12-19T09:13:27.512294)
  LIVETIME(2023-12-19T06:14:27.512294) ALTDATE(2023-12-20) ALTTIME(17.13.27)
  ```

  ```
  GRPNAME(beta) GRPROLE(Recovery) GRPADDR(9.20.135.130) GRPVER(09040300)
  CONNGRP(yes) GRSTATUS(Synchronizing) RCOVLSN(<0:0:19:16723>)
  RCOVTIME(2023-12-20T17:36:23.463209Z) BACKLOG(5619) INSYNC(no) SYNCTIME()
  ALTDATE(2023-12-20) ALTTIME(17.13.27)
  ```

Attività correlate

“Esempio: configurazione della HA nativa utilizzando IBM MQ Operator” a pagina 112
Questo esempio distribuisce un gestore code utilizzando la funzione di alta disponibilità nativa in OpenShift Container Platform utilizzando IBM MQ Operator. Il TLS reciproco viene utilizzato per l'autenticazione, per eseguire l'associazione da un certificato TLS a un'identità nel gestore code.

Riferimenti correlati

comando dspmq (visualizza gestori code)

OpenShift MQ Adv. ***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 100 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.

OpenShift MQ Adv. V 9.4.2 ***Messa a punto avanzata per Native HA CRR***

Impostazioni avanzate per la regolazione dei tempi e degli intervalli. Non dovrebbe essere necessario utilizzare queste impostazioni a meno che non si sappia che i valori predefiniti non corrispondono ai requisiti del sistema.

Le opzioni di base per la configurazione di Native HA CRR vengono gestite utilizzando l'API (`QueueManager`), che l' IBM MQ Operator e utilizza per configurare i file INI del gestore di code sottostante. Ci sono alcune opzioni più avanzate che sono configurabili solo utilizzando un file INI, sotto la sezione " NativeHARecoveryGroup " del file " qm.ini ". Vedi anche “Esempio: fornitura di file MQSC e INI” a pagina 100 per ulteriori informazioni su come configurare un file INI.

`ShortRetryTimer`
Specifica l'intervallo di attesa tra i tentativi di connessione a un gruppo di ripristino nella fase di breve riprova. Specificato in millisecondi, il valore predefinito è 2500 (2.5 i secondi).

`ShortRetryCount`
Specifica il numero di tentativi di connessione al gruppo di ripristino durante la breve fase di riprova.

`LongRetryTimer`
Specifica l'intervallo di attesa tra i tentativi di connessione a un gruppo di ripristino nella fase di ripetizione lunga. La lunga fase di riprova continua indefinitamente fino a quando non viene stabilita la connessione o fino a quando la configurazione non viene modificata per commentare o rimuovere l' `ReplicationAddress`. Specificato in millisecondi, il valore predefinito è 60000 (60 secondi).

`RetryReporting`
Impostare `Phase` o `All` per specificare se segnalare ogni tentativo di riprova o solo la fase (lunga o breve). Se non specificato, il valore predefinito è `Phase`.

`StatusInterval`
Specifica l'intervallo di tempo massimo che il gruppo di recupero attenderà prima di inviare uno stato del registro di recupero. Specificato in millisecondi, il valore predefinito è 10000 (10 secondi).

`StatusCommitThreshold`
Specifica il numero massimo di byte che il gruppo di ripristino conferma prima di inviare uno stato del registro di ripristino. Specificato in byte e predefinito a 0, il che significa che solo l' `StatusInterval` e viene utilizzato per determinare quando viene inviato lo stato del registro di ripristino.

OpenShift MQ Adv. ***Chiusura gestori code della HA nativa***

Per IBM MQ in container, è possibile utilizzare il comando **endmqm** per terminare un gestore di code attivo o una replica che fa parte di un gruppo Native HA.

Prima di iniziare

Nota: Queste informazioni si applicano solo agli ambienti container.

Informazioni su questa attività

La procedura di arresto di un gestore di code che fa parte di un gruppo Native HA dipende dal fatto che si tratti di un'istanza attiva o di una replica. Quando si termina uno dei due tipi di istanza, viene effettuato un controllo per garantire che la terminazione dell'istanza non interrompa il quorum del gruppo Native HA. Se il quorum viene meno, il comando **endmqm** fallisce.

Quando si impartisce un comando **endmqm**, le altre istanze del gruppo vengono avvertite che ciò sta accadendo, in modo che non segnalino errori quando la connessione si interrompe.

Se un'istanza attiva perde il quorum a causa dell'interruzione o della disconnessione di troppe istanze di replica, l'istanza attiva attende per un periodo di tempo configurabile prima di terminare completamente. In questo modo si ottiene un periodo di tempo per chiudere l'elaborazione con grazia, anziché interrompere le connessioni delle applicazioni. Questo valore di timeout può essere specificato dall'attributo `QuorumConnectivityTimeout` nella stanza `NativeHALocalInstance` del file `qm.ini`. Il valore predefinito è 0 secondi.

Procedura

- Per terminare l'istanza attiva di un gestore di code, eseguire il seguente comando sul nodo in cui è in esecuzione l'istanza attiva:

  ```
  endmqm -s QMgrName
  ```

  – Specificare l'opzione `-r` per aiutare le applicazioni client a riconnettersi a un'altra istanza.
  – Se l'istanza non è quella attiva nel gruppo Native HA, il comando fallisce.
  – Se l'interruzione di questa istanza attiva causerebbe il fallimento del quorum del gruppo, il comando fallisce. (Se altre istanze terminano o diventano non disponibili nello stesso momento in cui si esegue questo comando, il controllo del quorum potrebbe non rilevarlo, il gruppo Native HA termina e può essere riavviato solo quando sono disponibili abbastanza istanze)

  Quando il gestore di code attivo termina, una delle istanze di replica assume il ruolo attivo. Non è possibile specificare quale replica subentra, questo è determinato dalla negoziazione all'interno del gruppo e dipende da quale ha i registri delle transazioni più aggiornati.
- Per terminare un'istanza di replica di un gestore di code, eseguire il seguente comando:

  ```
  endmqm -x QMgrName
  ```

  – Se l'istanza è quella attiva, il comando fallisce.
  – Se l'interruzione di questa istanza di replica causerebbe il fallimento del quorum del gruppo, il comando fallisce. (Se altre istanze terminano o diventano non disponibili nello stesso momento in cui si esegue questo comando, il controllo del quorum potrebbe non rilevarlo, il gruppo Native HA termina e può essere riavviato solo quando sono disponibili abbastanza istanze)

  Nota: È inoltre possibile utilizzare gli switch -c, -i, -p o -w con il comando **endmqm** sulle istanze Native HA, indipendentemente dal ruolo ricoperto. L'istanza del gestore di code termina, ignorando l'effetto che ha sul quorum del gruppo. Le informazioni vengono comunque condivise con le altre istanze del gruppo. È possibile utilizzare questi interruttori insieme a -s per l'istanza attiva. Non è possibile utilizzare questi interruttori insieme all'interruttore -x per le istanze di replica.

Configurazione di un gestore code a più istanze utilizzando IBM MQ Operator

Questo esempio distribuisce un gestore code a più istanze utilizzando OpenShift Container Platform utilizzando IBM MQ Operator. Il TLS reciproco viene utilizzato per l'autenticazione, per eseguire l'associazione da un certificato TLS a un'identità nel gestore code.

Prima di iniziare

Per completare questo esempio, è necessario prima aver completato i prerequisiti riportati di seguito:

- Creare un progetto / spazio dei nomi 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 OpenShift Container Platform. Descrive inoltre i passi aggiuntivi richiesti per distribuire il gestore code con TLS abilitato.

Procedura

1. Determinare una classe di memoria adatta

   È possibile accedere all'archiviazione in un cluster Kubernetes utilizzando più modalità di accesso al volume persistente. Un gestore code a più istanze crea più volumi persistenti: uno per ciascun gestore code e almeno un volume condiviso. Il volume condiviso per un gestore code a più istanze deve utilizzare una classe di memorizzazione `ReadWriteMany` . La classe di archiviazione predefinita in un cluster Kubernetes è in genere per una classe di archiviazione `ReadWriteOnce` (archiviazione blocchi). Ad esempio, se si utilizza Red Hat OpenShift Data Foundation, la classe di archiviazione *ocs - storagecluster - cephfs* fornisce un file system condiviso adatto. La scelta del file system è molto importante, perché non tutti i file system condivisi gestiscono il blocco dei file nello stesso modo. Consultare Planning file system support on Multiplatforms e Test statement for IBM MQ multi - instance queue manager file systems.
2. Creare una coppia di certificati come descritto in “Creazione di una PKI autofirmata utilizzando OpenSSL” a pagina 104.
3. Crea una mappa di configurazione contenente comandi MQSC e un file INI

   Creare una Kubernetes ConfigMap 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.

   Assicurati di essere nello spazio dei nomi che hai creato precedentemente (vedi Prima di iniziare), quindi immetti il seguente YAML nella console web OCP o utilizzando la riga di comando.

   ```
   apiVersion: v1
   kind: ConfigMap
   metadata:
     name: example-miqm-configmap
   data:
     example-tls.mqsc: |
       DEFINE CHANNEL('MTLS.SVRCONN') CHLTYPE(SVRCONN) SSLCAUTH(REQUIRED)
   SSLCIPH('ANY_TLS13_OR_HIGHER') REPLACE
       SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=*') USERSRC(NOACCESS)
   ACTION(REPLACE)
       SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=example-app1') USERSRC(MAP)
   MCAUSER('app1') ACTION(REPLACE)
       SET AUTHREC PRINCIPAL('app1') OBJTYPE(QMGR) AUTHADD(CONNECT,INQ)
       DEFINE QLOCAL('EXAMPLE.QUEUE') REPLACE
       SET AUTHREC PROFILE('EXAMPLE.QUEUE') PRINCIPAL('app1') OBJTYPE(QUEUE)
   AUTHADD(BROWSE,PUT,GET,INQ)
     example-tls.ini: |
       Service:
           Name=AuthorizationService
           EntryPoints=14
           SecurityPolicy=UserExternal
   ```

   MQSC definisce un canale denominato *MTLS.SVRCONN* e una coda denominata *EXAMPLE.QUEUE*. Il canale è configurato per consentire l'accesso solo ai client che presentano un certificato con un "nome comune" *example-app1*. Questo è il nome comune utilizzato in uno dei certificati creati nel passo “2” a pagina 146. Le connessioni su questo canale con questo nome comune vengono associate a un ID utente *app1*, autorizzato a connettersi al gestore code e ad accedere alla coda di esempio. Il file INI abilita una politica di sicurezza che indica che l'ID utente *app1* non deve necessariamente esistere in un registro utente esterno - esiste solo come nome in questa configurazione.
4. Distribuisci il gestore code

   Creare un nuovo gestore code utilizzando la seguente risorsa personalizzata YAML. Assicurarsi di essere nello spazio dei nomi creato prima di iniziare questa operazione, quindi inserire il seguente YAML nella console web di OCP o tramite la riga di comando. Verificare che sia stata specificata la licenza corretta e accettarla modificando `false` in `true`.

Nota: Questo esempio utilizza una licenza IBM MQ Advanced , ma è possibile utilizzare anche una licenza IBM MQ . Per ulteriori informazioni, consultare "Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator" a pagina 160.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: exampleqm
spec:
  license:
    accept: false
    license: L-NUUP-23NH8Y
    use: Production
  queueManager:
    name: EXAMPLEQM
    availability:
      type: MultiInstance
    mqsc:
    - configMap:
        name: example-miqm-configmap
        items:
        - example-tls.mqsc
    ini:
    - configMap:
        name: example-miqm-configmap
        items:
        - example-tls.ini
    storage:
      defaultClass: STORAGE_CLASS
  version: 9.4.3.0-r1
  pki:
    keys:
      - name: default
        secret:
          secretName: example-qm-tls
          items:
            - tls.key
            - tls.crt
            - ca.crt
```

Modificare *STORAGE_CLASS* nella classe di memoria identificata nel Passo "1" a pagina 146.

Nota che il segreto *example - qm - tls* è stato creato al passo "2" a pagina 146e che ConfigMap *example - miqm - configmap* è stato creato al passo "3" a pagina 146

Il tipo di disponibilità è impostato su *MultiInstance*, che fa sì che lo storage persistente venga selezionato automaticamente.

5. 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 exampleqm
   ```

6. Verifica la connessione al gestore code

   Per confermare che il gestore code è configurato e disponibile, effettuare le operazioni riportate in "Verifica di una connessione TLS reciproca a un gestore code dal tuo laptop" a pagina 108.

7. Forza l'esito negativo del pod attivo

   Per convalidare il ripristino automatico del gestore code, simula un errore pod:

   a) Visualizza i pod attivi e in standby

      Esegui il seguente comando:

      ```
      oc get pods --selector app.kubernetes.io/instance=exampleqm
      ```

      Tieni presente che, nel campo **READY** , il pod attivo restituisce il valore `1/1`, mentre il pod standby restituisce il valore `0/1`.

   b) Elimina il pod attivo

      Esegui il seguente comando, specificando il nome completo del pod attivo:

```
oc delete pod exampleqm-ibm-mq-value
```

c) Visualizza di nuovo lo stato del pod

Esegui il seguente comando:

```
oc get pods --selector app.kubernetes.io/instance=exampleqm
```

d) Visualizza lo stato del gestore code

Esegui il seguente comando, specificando il nome completo dell'altro pod:

```
oc exec -t Pod -- dspmq -x
```

Dovresti vedere lo stato che mostra che l'istanza attiva è cambiata, ad esempio:

```
QMNAME(EXAMPLEQM)                                         STATUS(Running as standby)
    INSTANCE(exampleqm-ibm-mq-1) MODE(Active)
    INSTANCE(exampleqm-ibm-mq-0) MODE(Standby)
```

e) Verificare nuovamente la connessione al gestore code

Per confermare che il gestore code è stato ripristinato, effettuare le operazioni riportate in “Verifica di una connessione TLS reciproca a un gestore code dal tuo laptop” a pagina 108.

Risultati

Congratulazioni, hai distribuito correttamente un gestore code a più istanze con autenticazione TLS reciproca e hai verificato che venga ripristinato automaticamente quando il pod attivo ha esito negativo.

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

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.

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.

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

Il client IBM MQ gestito .NET imposta SERVERNAME sul rispettivo nome host se la proprietà **OutboundSNI** è impostata su HOSTNAME, che consente a un client IBM MQ gestito .NET di connettersi a un gestore code utilizzando gli instradamenti Red Hat OpenShift .

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: unique_name_for_the_route
  namespace: namespace_of_your_MQ_deployment
spec:
```

```
  host: SNI_address_mapping_for_the_channel
  to:
    kind: Service
    name: name_of_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 217
Come connettersi al IBM MQ Console di un gestore code distribuito su un cluster Red Hat OpenShift Container Platform .

Configurazione della registrazione di audit per IBM MQ nei container

IBM MQ nei contenitori fornisce varie strutture per la registrazione di audit di diversi eventi.

A seconda dei requisiti di registrazione degli audit per il vostro ambiente, potrebbe essere necessario configurare una o più di queste strutture e meccanismi adeguati per la raccolta e la conservazione dei registri degli eventi. Scegli dalle seguenti opzioni.

IBM MQ Operator convalida del webhook

A partire dalla versione 3.6.0, per impostazione predefinita, IBM MQ Operator invia gli eventi di audit degli accessi agli endpoint esposti al log del contenitore del pod. Non è richiesta alcuna configurazione.

IBM MQ eventi

È possibile configurare IBM MQ per produrre messaggi di evento a scopo di auditing. Per ulteriori informazioni, vedere Audit.

V 9.4.3 MQ metriche del contenitore

Il contenitore IBM MQ espone un endpoint di metriche Prometheus (per ulteriori informazioni, vedere “Monitoraggio quando si utilizza IBM MQ Operator” a pagina 217). Da IBM MQ 9.4.3, è possibile abilitare la registrazione degli accessi a questo endpoint impostando la variabile d'ambiente `MQ_LOGGING_METRICS_AUDIT_ENABLED` su `true`. Se si utilizza IBM MQ Operator, si può impostare questa variabile nella sezione `env` della configurazione di `queueManager` :

```
[...]
spec:
  queueManager:
    env:
    - name: MQ_LOGGING_METRICS_AUDIT_ENABLED
      value: "true"
[...]
```

Quando la registrazione è abilitata, i file di log vengono generati nella directory `/var/mqm/errors` . È possibile persistere i dati presenti in questa directory montando uno storage esterno su `/mnt/mqm` (vedere il riferimento all'immagine del contenitore). Se si utilizza il sito IBM MQ Operator, è possibile invece mappare l'archiviazione esterna utilizzando i campi “.spec.queueManager.storage.persistedData” a pagina 266 . I file di log degli accessi alle metriche sono scritti in formato JSON. I nomi dei file hanno la forma `metricaudit<nn>.json`, dove nn è un numero. Il file `metricaudit01.json` contiene gli eventi più recenti.

Webserver incorporato IBM WebSphere Application Server Liberty

È possibile configurare il webserver incorporato WebSphere Liberty per produrre eventi di audit. Le funzioni `audit-1.0` e `audit-2.0` sono installate e disponibili per la configurazione, a seconda delle esigenze. Per ulteriori informazioni, vedere Audit 1.0 e Audit 2.0 nella documentazione di WebSphere Liberty .

È possibile attivare e configurare la funzione utilizzando un file di configurazione. Se si utilizza IBM MQ Operator, è possibile caricare il file utilizzando i supporti “.spec.web.manualConfig” a pagina 275 . Ad esempio, il codice seguente abilita la configurazione predefinita della funzione Audit 2.0.

Creare un file di configurazione chiamato `mqwebuser.xml`, per abilitare la funzione `audit-2.0` :

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: example-enable-audit
data:
  mqwebuser.xml: |
    <?xml version="1.0" encoding="UTF-8"?>
    <server>
      <featureManager>
        <feature>audit-2.0</feature>
      </featureManager>
    </server>
```

Aggiungere il codice precedente alla configurazione di `queueManager` usando il mount `spec.web.manualConfig` :

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
[...]
spec:
  [...]
  web:
    enabled: true
    manualConfig:
      configMap:
        name: example-enable-audit
```

Nota: Per impostazione predefinita, i registri di audit vengono rispecchiati nel registro del contenitore in formato base (testo semplice). Questo formato contiene una serie ridotta di informazioni di audit. Per ottenere maggiori informazioni, modificare il log del contenitore IBM MQ in formato JSON o disabilitare il mirroring del log web e raccogliere direttamente il file di log di audit JSON di WebSphere Liberty . È possibile controllare il formato dei registri e le fonti di mirroring dei registri usando le variabili d'ambiente `MQ_LOGGING_CONSOLE_FORMAT` e `MQ_LOGGING_CONSOLE_SOURCE` . Per i valori validi, vedere “Variabili d'ambiente supportate” a pagina 288.

OpenShift CP4I Integrazione di IBM MQ con la traccia IBM Instana

IBM Instana può essere utilizzato per tracciare le transazioni in IBM Cloud Pak for Integration.

Prima di iniziare

Questo documento riguarda la traccia IBM Instana , che è il processo di traccia dei messaggi attraverso un sistema. Non copre il monitoraggio IBM Instana , in cui vengono richiamati i dettagli sullo stato di un gestore code IBM MQ . Per informazioni sul controllo di IBM MQ da parte di IBM Instana , consultare Monitoraggio IBM MQ. Per istruzioni dettagliate sul monitoraggio autenticato, consultare “Configurazione del monitoraggio di IBM Instana autenticato con TLS” a pagina 153.

Nota:

- Questa funzione è supportata solo dagli operandi di IBM MQ versione 9.3.1.0-r2 o successiva.
- È possibile eseguire il tracciamento di IBM Instana sulle versioni precedenti di IBM MQ Operator e del gestore di code, ma non in modo nativo. Vedere Configurazione del tracciamento di IBM MQ nella documentazione di IBM Instana .

- Questa funzione non è supportata quando si utilizza la rete IPv6 .

Prima di poter eseguire la traccia di IBM Instana con l'operatore IBM MQ , è necessario distribuire sia un backend IBM Instana che agent IBM Instana . Per impostazione predefinita, un gestore code IBM MQ comunica con un agente IBM Instana distribuito sullo stesso nodo del pod del gestore code.

Informazioni su questa attività

L'abilitazione dell'integrazione con IBM Instana comporta l'installazione di un'uscita API IBM MQ nel tuo gestore code. L'uscita API invia i dati di traccia agli agent IBM Instana relativi ai messaggi che passano attraverso il gestore code.

L'uscita API aggiunge intestazioni RFH2 a ciascun messaggio. Queste intestazioni contengono informazioni di traccia.

Gli agent IBM Instana sono responsabili dell'invio dei dati di traccia al backend IBM Instana .

Per informazioni sulla distribuzione di un backend IBM Instana e di agent IBM Instana , vedi Abilitazione dei link di monitoraggio Instana nell'IU della piattaforma nella documentazione IBM Instana .

Procedura

Distribuzione standard

- Distribuire un gestore code con la traccia IBM Instana abilitata.

  Per impostazione predefinita, la traccia IBM Instana è disabilitata.

  Se si sta utilizzando la IBM Cloud Pak for Integration Platform UI o la console Web OpenShift :

  1. Fare clic su **Operatori** > **Operatori installati** > **IBM MQ** > **Dettagli** > **Crea istanza**. Fornite tutti i dettagli necessari al vostro gestore di code.
  2. Nella stessa posizione, fare clic su **Telemetria** > **Tracing** > **Instana**.
  3. Imposta l'opzione **Abilita traccia Instana** su `true`.

  Se si sta eseguendo la distribuzione tramite YAML, utilizzare il seguente frammento:

```
spec:
  telemetry:
    tracing:
      instana:
        enabled: true
```

Distribuzione avanzata

- Comunicare con l'agent IBM Instana su https.

  Per impostazione predefinita, l'uscita IBM Instana per IBM MQ comunica con IBM Instana Agent su http. L'indirizzo host dell'agent è impostato sull'indirizzo IP del nodo su cui è in esecuzione il gestore code. Ciò corrisponde alla configurazione descritta in Abilitazione del monitoraggio IBM Instana nella documentazione IBM Instana , dove gli agent IBM Instana vengono distribuiti dall'operatore agent IBM Instana come una serie di daemon.

  Attualmente la comunicazione tra l'uscita IBM Instana per IBM MQ e l'agente IBM Instana supporta i protocolli http o https. Per utilizzare https, l'agent IBM Instana deve essere prima configurato per utilizzare la codifica TLS. Vedi Impostazione della codifica TLS per l'endpoint dell'agent nella documentazione di IBM Instana . Il protocollo può quindi essere impostato su https nel modo seguente:

  Se stai utilizzando la console web OpenShift :

  1. Fare clic su **Operatori** > **Operatori installati** > **IBM MQ** > **Dettagli** > **Crea istanza**. Fornite tutti i dettagli necessari al vostro gestore di code.
  2. Nella stessa posizione, fare clic su **Telemetria** > **Instana**.
  3. Espandere l'elenco a discesa **Configurazione avanzata** .

4. Impostare il **protocollo di comunicazione agent Instana** su `https`.

Se si sta eseguendo la distribuzione tramite YAML, utilizzare il seguente frammento:

```
spec:
  telemetry:
    instana:
      enabled: true
      protocol: https
```

- Impostare **agentHost**

  Se gli agent IBM Instana non sono stati distribuiti come serie di daemon sul cluster Openshift in cui è in esecuzione il gestore code, è necessario impostare il valore **agentHost** sul nome host o sull'indirizzo IP in cui è in esecuzione l'agent IBM Instana . Il valore **agentHost** non deve includere un protocollo o una porta.

  Se stai utilizzando la console web OpenShift :

  1. Fare clic su **Operatori** > **Operatori installati** > **IBM MQ** > **Dettagli** > **Crea istanza**. Fornite tutti i dettagli necessari al vostro gestore di code.
  2. Nella stessa posizione, fare clic su **Telemetria** > **Instana**.
  3. Espandere l'elenco a discesa **Configurazione avanzata** .
  4. Immetti il nome host nella casella di testo **Instana agent host** .

  Se si sta eseguendo la distribuzione tramite YAML, utilizzare il seguente frammento:

```
spec:
  telemetry:
    instana:
      enabled: true
      agentHost: 9.9.9.9
```

Operazioni successive

Consultare anche “Distribuzione di un gestore code semplice utilizzando IBM MQ Operator” a pagina 98.

OpenShift CP4I Operator2.2.0 Configurazione del monitoraggio di IBM Instana autenticato con TLS

Per poter monitorare un gestore code tramite IBM Instana , è necessario configurare sia l'agent che il gestore code.

Prima di iniziare

La sezione "Configurazione" di "Monitoraggio IBM MQ" nella documentazione IBM Instana fornisce informazioni generali relative alla configurazione del controllo IBM Instana . Tuttavia, non include dettagli sulla configurazione del gestore code.

Prima di poter eseguire la traccia di IBM Instana con l'operatore IBM MQ , è necessario distribuire sia un backend IBM Instana che agent IBM Instana . Per farlo, vedi Abilitazione del monitoraggio di IBM Instana nella CP4I Platform UI nella documentazione di IBM Instana .

Procedura

1. Genera certificati.
2. Configurare gli agent IBM Instana.
3. Configurazione del gestore code.
4. Verifica e debug.

Attività correlate

“Integrazione di IBM MQ con la traccia IBM Instana” a pagina 151

IBM Instana può essere utilizzato per tracciare le transazioni in IBM Cloud Pak for Integration.

OpenShift CP4I Operator2.2.0 *Generare un certificato e una chiave per l'agent IBM Instana e il gestore code*

Per le comunicazioni TLS tra l'agente IBM Instana e il gestore code, entrambi devono avere un certificato e la chiave privata corrispondente.

Prima di iniziare

Questa è la prima di quattro attività per configurare il monitoraggio IBM Instana autenticato con TLS.

Nota: I valori utilizzati nella creazione di questi certificati sono a scopo dimostrativo. Durante la distribuzione in un ambiente di produzione, verificare che l'oggetto e la scadenza del certificato siano appropriati.

Procedura

IBM MQ Gestore code

Per comunicare con l'agent IBM Instana tramite TLS, il gestore code deve avere un certificato e la chiave privata corrispondente. Se hai già questi, salta questa sezione.

1. Generare un certificato e una chiave privata per il gestore code.

   Esegui il seguente comando:

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

AgenteIBM Instana

Per consentire all'agent di eseguire la comunicazione TLS con il gestore code IBM MQ , l'agent deve avere un certificato e la chiave privata corrispondente. Se si dispone già di una chiave privata e di un certificato in un keystore JKS che si desidera utilizzare, ignorare questa sezione.

2. Generare un certificato e una chiave privata per l'agent IBM Instana .

   Esegui il seguente comando:

   ```
   openssl req \
          -newkey rsa:2048 -nodes -keyout application.key \
          -subj "/CN=instana-agent/OU=app team1" \
          -x509 -days 3650 -out application.crt
   ```

3. Memorizzare il certificato e la chiave privata in un keystore PKCS12 .

   Esegui il seguente comando, sostituendo *your_password* con la password che vuoi utilizzare per proteggere il keystore. Eseguire questa sostituzione in tutte le operazioni successive.

   ```
   openssl pkcs12 -export -out application.p12 -inkey application.key -in application.crt
   -passout pass:your_password
   ```

4. Convertire il keystore PKCS12 in un keystore JKS.

   Esegui il seguente comando:

   ```
   keytool -importkeystore \
           -srckeystore application.p12 \
           -srcstoretype pkcs12 \
           -destkeystore application.jks \
           -deststoretype JKS \
           -srcstorepass your_password \
           -deststorepass your_password \
           -noprompt
   ```

5. Etichettare il certificato.

Esegui il seguente comando:

```
keytool -changealias -alias "1" -destalias "instana" -keypass your_password -keystore
application.jks -storepass your_password -noprompt
```

6. Importare il certificato del gestore code nel keystore.

   Esegui il seguente comando:

```
keytool -importcert -file server.crt -keystore application.jks -storepass your_password
-alias myca -noprompt
```

Operazioni successive

Ora è possibile configurare gli agenti per il IBM Instana controllo.

Monitoraggio Instana: configurazione degli agent

Montare il keystore sugli agent IBM Instana , quindi configurare il controllo per un gestore code specifico.

Prima di iniziare

Questa attività presuppone che sia stato generato un certificato e una chiave per gli agent IBM Instana e il gestore code.

Procedura

Montaggio del keystore sugli agenti IBM Instana

1. Creare un segreto dal keystore JKS nello spazio dei nomi dell'agente IBM Instana .

   Esegui il seguente comando, sostituendo *keystore_secret_name* col nome che vuoi utilizzare. Eseguire questa sostituzione in tutte le operazioni successive.

```
oc create secret generic keystore_secret_name --from-file=./application.jks -n instana-agent
```

2. Montare il segreto come un volume nel pod dell'agente Instana.

   Per farlo, specificare il volume e i supporti del volume nell' `CustomResource` e dell'agente o nella tabella Helm dell' `values.yaml` :

```
agent:
  pod:
    volumeMounts:
    - mountPath: /opt/instana/agent/etc/jks-file-name.jks
      name: mq-key-jks-name
      subPath: jks-file-name.jks
    volumes:
    - name: mq-key-jks-name
      secret:
        secretName: keystore_secret_name
```

   Per apportare questa modifica a un agente Instana distribuito dall'operatore dell'agente Instana, eseguire un comando simile al seguente:

```
oc edit agent [instana-agent-name] -n [instana-agent-namespace]
```

   Per ulteriori informazioni sui segreti di montaggio nel pod dell'agente Instana, consultare Volumi e volumeMounts nella documentazione della tabella dell'agente Instana (Helm).

Configurazione del monitoraggio per un determinato gestore code

3. Nello spazio dei nomi instana - agent, utilizza il comando `oc edit configmap instana-agent` per modificare la mappa di configurazione instana - agent.
4. Aggiungere la seguente sezione in `configuration.yaml: |`. Se questa sezione è già stata definita, è sufficiente aggiungere il nuovo gestore code all'elenco.

```
    com.instana.plugin.ibmmq:
      enabled: true
      poll_rate: 60
      queueManagers:
         QUEUE_MANAGER_NAME:
            channel: 'INSTANA.A.SVRCONN'
            keystorePassword: 'your_password'
            keystore: '/opt/instana/agent/etc/application.jks'
            cipherSuite: 'TLS_RSA_WITH_AES_256_CBC_SHA256'
```

Dove

- *your_password* è la parola d'ordine per il keystore JKS
- *QUEUE_MANAGER_NAME* è il nome del gestore code IBM MQ sottostante da distribuire, anziché il nome dell'operatore del gestore code.

Nota: Se *QUEUE_MANAGER_NAME* non è impostato sul nome del gestore code sottostante ed è invece impostato su Operando, il controllo non funzionerà. Il nome sottostante è definito in `spec.queuemanager.name` per l'operatore del gestore code.

5. Elimina i pod instana - agent nello spazio nomi instana - agent. Ciò li fa riavviare e iniziare il controllo con le nuove impostazioni.

Operazioni successive

Si è ora pronti a configurare il gestore code per il monitoraggio IBM Instana.

OpenShift CP4I Operator2.2.0 *Monitoraggio Instana: configurazione del gestore code*

Configurare un gestore code che utilizza TLS per comunicare con l'agent IBM Instana .L'autenticazione per questa connessione viene eseguita utilizzando un SSLPEERMAP.

Prima di iniziare

Questa attività presuppone che l'utente abbia configurato gli agenti per il IBM Instana monitoraggio.

Procedura

1. Configurare il gestore code tramite MQSC e INI.

   MQSC viene utilizzato per impostare un nuovo canale abilitato TLS e quindi configurare tale canale per autenticare l'agent IBM Instana di connessione se dispone di un certificato con i campi richiesti. In questo caso, associamo qualsiasi client di connessione con un certificato che contiene i campi `CN=instana-agent,OU=app team1` all'utente `app1`. MQSC, quindi, concede l'autorizzazione all'utente `app1` per eseguire le operazioni richieste per il monitoraggio IBM Instana .

   Il file INI viene utilizzato per concedere autorizzazioni all'utente esterno `app1`.

   La seguente configmap contiene le impostazioni MQSC e INI richieste. Distribuiscila nel tuo spazio dei nomi del gestore code.

```
apiVersion: v1
data:
  channel.mqsc: |-
    DEFINE CHANNEL('INSTANA.A.SVRCONN') CHLTYPE(SVRCONN) SSLCAUTH(REQUIRED)
SSLCIPH('ANY_TLS12_OR_HIGHER')
    ALTER QMGR CONNAUTH(' ')
    REFRESH SECURITY
    SET CHLAUTH('INSTANA.A.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=*') USERSRC(NOACCESS)
ACTION(REPLACE)
    SET CHLAUTH('*') TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(NOACCESS) ACTION(REPLACE)
    SET CHLAUTH('INSTANA.A.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=instana-agent,OU=app
team1') USERSRC(MAP) MCAUSER('app1')
    SET AUTHREC PRINCIPAL('app1') OBJTYPE(QMGR) AUTHADD(ALL)
    SET AUTHREC PROFILE('SYSTEM.ADMIN.COMMAND.QUEUE') PRINCIPAL('app1') OBJTYPE(QUEUE)
AUTHADD(PUT,INQ,DSP,CHG)
    SET AUTHREC PROFILE('SYSTEM.**') PRINCIPAL('app1') OBJTYPE(TOPIC) AUTHADD(DSP)
    SET AUTHREC PROFILE('*') PRINCIPAL('app1') OBJTYPE(TOPIC) AUTHADD(DSP)
```

```
    SET AUTHREC PROFILE('SYSTEM.**') PRINCIPAL('app1') OBJTYPE(QUEUE) AUTHADD(DSP, CHG, GET)
    SET AUTHREC PROFILE('SYSTEM.**') PRINCIPAL('app1') OBJTYPE(LISTENER) AUTHADD(DSP)
    SET AUTHREC PROFILE('AMQ.*') PRINCIPAL('app1') OBJTYPE(QUEUE) AUTHADD(DSP, CHG)
    REFRESH SECURITY TYPE(CONNAUTH)
  auth.ini: |-
    Service:
      Name=AuthorizationService
      EntryPoints=14
      SecurityPolicy=UserExternal
kind: ConfigMap
metadata:
  namespace: your-queue-manager-namespace
  name: qmgr-monitoring-config
```

dove *your - queue - manager - namespace* è lo spazio dei nomi in cui verrà distribuito il gestore code.

Nota: Se si stanno monitorando code definite dall'utente, è necessario aggiungere ulteriori righe alla configmap MQSC, concedendo le autorizzazioni DSP, CHG e GET a tali code. Ad esempio:

```
SET AUTHREC PROFILE('MYQUEUE') PRINCIPAL('app1') OBJTYPE(QUEUE) AUTHADD(DSP, CHG, GET).
```

Questo esempio utilizza una configmap per i dati MQSC e INI, ma è possibile utilizzare un segreto se eventuali aggiunte effettuate sono riservate. Per informazioni generali sulla distribuzione con MQSC e INI, consultare “Esempio: fornitura di file MQSC e INI” a pagina 100.

2. Per stabilire una connessione TLS, il gestore code deve considerare attendibile il certificato dell'agent IBM Instana . Per ottenere ciò, creare un segreto contenente solo il certificato dell'agent IBM Instana :

```
oc create secret generic instana-certificate-secret --from-file=./application.crt -n your-
queue-manager-namespace
```

3. Il gestore code deve presentare il proprio certificato per l'handshake TLS e richiede l'accesso alla chiave privata associata. Distribuisci un segreto contenente la chiave e il certificato che hai creato in precedenza o che già possiedi:

```
oc create secret tls qm-tls-secret --cert server.crt --key server.key -n your-queue-manager-
namespace
```

Con la configmap e il segreto creati, si è pronti a creare il gestore code stesso.

4. Assicurarsi che il gestore code YAML non imposti la variabile di ambiente **MQSNOAUT** nel contenitore del gestore code.

   In caso contrario, una volta abilitato, il meccanismo di autenticazione non funzionerà. La rimozione della variabile dopo la distribuzione non comporta la riabilitazione del meccanismo e la ricreazione del gestore code.

5. Aggiungere le seguenti sezioni alla definizione del gestore code, dove *MYQM* è il nome del gestore code:

```
spec:
  queueManager:
    name: MYQM   #(a)
    ini:   #(b)
      - configMap:
          items:
            - auth.ini
          name: qmgr-monitoring-config
    mqsc:   #(c)
      - configMap:
          items:
            - channel.mqsc
          name: qmgr-monitoring-config
  pki:
    keys:   #(d)
        - name: default
          secret:
            items:
            - tls.key
            - tls.crt
            secretName: qm-tls-secret
    trust:   #(e)
      - name: app
```

```
        secret:
          items:
            - application.crt
          secretName: instana-certificate-secret
```

Le sezioni contrassegnate della specifica sono descritte come segue:

a. Assicurarsi di aver fornito al gestore code sottostante un nome univoco. Se il gestore code sottostante non ha un nome univoco, il monitoraggio potrebbe non funzionare come previsto. Questo nome deve corrispondere al nome nella mappa di configurazione dell'agent IBM Instana che è stata modificata in precedenza.

b. Le informazioni INI scritte nella configmap vengono aggiunte al gestore code.

c. Le informazioni MQSC scritte nella configmap vengono aggiunte al gestore code.

d. Il certificato del gestore code e la chiave privata vengono aggiunti al keystore del gestore code.

e. Il certificato dell'agent IBM Instana viene aggiunto al truststore del gestore code.

6. Opzionale: Abilitare IBM Instana Traccia sul gestore code monitorato.

   Se si desidera eseguire questa operazione, consultare “Integrazione di IBM MQ con la traccia IBM Instana” a pagina 151.

7. Distribuire il gestore code.

Operazioni successive

Sei ora pronto a verificare ed eseguire il debug del IBM Instana monitoraggio.

OpenShift CP4I Operator2.2.0 *Monitoraggio Instana: verifica e debug*

Per poter monitorare un gestore code tramite IBM Instana , è necessario configurare sia l'agent che il gestore code.

Prima di iniziare

Questa attività presuppone che sia stato configurato il gestore code per il monitoraggio IBM Instana.

Procedura

Verifica

1. Per verificare che la distribuzione sia stata eseguita correttamente, visualizzare il gestore code nel dashboard IBM Instana .

   Il gestore code deve essere visibile nella sezione dei servizi della pagina dell'applicazione e anche nella vista Infrastruttura.

Debug

Nota: Questi passi di debug presuppongono una distribuzione Openshift dell'agent IBM Instana in esecuzione come serie di daemon.

Se non è possibile visualizzare il gestore code nel dashboard IBM Instana , è possibile che il gestore code sia stato configurato in modo non corretto. Utilizzare la seguente procedura per indagare.

2. Identificare il nodo su cui è in esecuzione il pod del gestore code attivo.

   Immettere il seguente comando nello spazio dei nomi del gestore code:

   ```
   oc get pods -o wide -n your-queue-manager-namespace
   ```

3. Per determinare quale pod dell'agent IBM Instana è in esecuzione sullo stesso nodo del tuo gestore code, esegui lo stesso comando nello spazio dei nomi instana - agent:

   ```
   oc get pods -o wide -n instana-agent-namespace
   ```

4. Per facilitare la comprensione di eventuali problemi dal lato dell'agent IBM Instana , ottenere i log del pod dell'agent IBM Instana e cercare le voci relative a 'mq' o al nome del gestore code.

Esegui il seguente comando:

```
oc logs instana-agent-pod -c instana-agent -n instana-agent
```

5. Controllare i log del gestore code.

   Se l'agent ha effettuato un tentativo di connessione al gestore code, i log del gestore code dovrebbero indicare il motivo per cui la connessione non è riuscita. Esegui il seguente comando:

   ```
   oc logs your-queue-manager-name -n your-queue-manager-namespace
   ```

Risultati

Sono state completate tutte e quattro le attività per configurare il monitoraggio IBM Instana autenticato con TLS.

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 Crea il segreto della chiave di titolarità.

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:9.4.3.0-r1
         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:9.4.3.0-r1"
      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 5 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 semplice utilizzando IBM MQ Operator” a pagina 98, 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 semplice utilizzando IBM MQ Operator” a pagina 98
Questo esempio distribuisce un gestore code "quick start", che utilizza la memoria effimera (non persistente) e disattiva la sicurezza IBM MQ . I messaggi non vengono resi persistenti durante i riavvii del gestore code. È possibile modificare la configurazione per modificare molte impostazioni del gestore code.

OpenShift CP4I Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator

Le annotazioni devono essere applicate ai gestori di code per configurarli con una licenza IBM MQ .

Prima di iniziare

Nota: Queste istruzioni sono valide solo se si utilizza il sito IBM MQ Operator. Se si stanno distribuendo i gestori di code su Kubernetes, vedere invece le annotazioni sulla licenza del contenitore IBM MQ.

Informazioni su questa attività

È possibile aggiungere annotazioni ai gestori di code sia quando si creano i gestori di code, sia aggiornando i gestori di code esistenti attraverso il sito `spec.annotations`.

Importante: L'aggiornamento dello YAML del gestore delle code provoca il riavvio del pod.

Procedura

Aggiornare le annotazioni di licenza dei gestori di code esistenti seguendo le istruzioni riportate in "Aggiungere annotazioni ai gestori di code utilizzando l'opzione IBM MQ Operator" a pagina 162, utilizzando le annotazioni di licenza richieste per ciascun gestore di code, in base alla propria abilitazione.

Annotazioni richieste per la licenza

IBM MQ

```
spec:
  annotations:
    productID: c661609261d5471fb4ff8970a36bccea
    productName: IBM MQ
```

IBM MQ for Non-Production Environment

```
spec:
  annotations:
    productID: 151bec68564a4a47a14e6fa99266deff
    productName: IBM MQ for Non-Production Environment
```

IBM MQ Contenitore multi istanza

```
spec:
  annotations:
    productID: 2dea73b866b648b6b4abe2a85eb76964
    productName: IBM MQ Container Multi Instance
```

IBM MQ Container multi istanza per ambienti non di produzione:

```
spec:
  annotations:
    productID: af11b093f16a4a26806013712b860b60
    productName: IBM MQ Container Multi Instance for Non-Production Environment
```

IBM MQ con CP4I diritto

```
spec:
  annotations:
    cloudpakId: c8b82d189e7545f0892db9ef2731b90d
    cloudpakName: IBM Cloud Pak for Integration
    productID: c661609261d5471fb4ff8970a36bccea
    productName: IBM MQ
    productMetric: VIRTUAL_PROCESSOR_CORE
    productCloudpakRatio: 4:1
```

IBM MQ per l'ambiente non di produzione con diritto a CP4I

```
spec:
  annotations:
    cloudpakId: c8b82d189e7545f0892db9ef2731b90d
    cloudpakName: IBM Cloud Pak for Integration
    productID: 151bec68564a4a47a14e6fa99266deff
    productName: IBM MQ for Non-Production Environment
    productMetric: VIRTUAL_PROCESSOR_CORE
    productCloudpakRatio: 8:1
```

IBM MQ Container Multi Istanza con diritto a CP4I

Per :

```
spec:
  annotations:
    productName: IBM MQ Container Multi Instance
    productID: 2dea73b866b648b6b4abe2a85eb76964
    productMetric: VIRTUAL_PROCESSOR_CORE
    productCloudpakRatio: 10:3
    cloudpakName: IBM Cloud Pak for Integration
    cloudpakId: c8b82d189e7545f0892db9ef2731b90d
```

IBM MQ Container Multi Istanza per ambiente non di produzione con diritti CP4I

```
spec:
  annotations:
    cloudpakId: c8b82d189e7545f0892db9ef2731b90d
    cloudpakName: IBM Cloud Pak for Integration
    productID: af11b093f16a4a26806013712b860b60
    productName: IBM MQ Container Multi Instance for Non-Production Environment
    productMetric: VIRTUAL_PROCESSOR_CORE
    productCloudpakRatio: 20:3
```

OpenShift CP4I Aggiungere annotazioni ai gestori di code utilizzando l'opzione IBM MQ Operator

È possibile aggiungere annotazioni ai gestori di code sia al momento della creazione, sia aggiornando i gestori di code esistenti utilizzando il sito `spec.annotations`.

Informazioni su questa attività

Queste istruzioni si applicano solo se entrambe le seguenti affermazioni sono vere:

- State utilizzando il sito IBM MQ Operator. Se state distribuendo i gestori di code su Kubernetes, consultate invece “IBM MQ annotazioni sulla licenza del contenitore” a pagina 281 .
- Si sta modificando un gestore di code esistente. Per i nuovi gestori di code, aggiungere invece le annotazioni alla creazione.

Importante: L'aggiornamento dello YAML del gestore delle code provoca il riavvio del pod.

Procedura

- **Opzione 1:** Aggiungere annotazioni utilizzando la console OpenShift .
  a) Andare al proprio gestore di code
    a. Accedere alla console di OpenShift con le credenziali di amministratore del cluster Red Hat OpenShift Container Platform .
    b. Cambiare il **progetto** nello spazio dei nomi in cui è stato installato IBM MQ Operator. Selezionare lo spazio dei nomi dall'elenco a discesa **Progetto**.
    c. Nel riquadro di navigazione, fare clic su **OperatoriOperatori** > **installati**.
    d. Nell'elenco del pannello Operatori installati, individuare e fare clic su **IBM MQ**.
    e. Fare clic sulla scheda **Queue Manager**.
    f. Fare clic sul **nome del gestore code** desiderato.
  b) Aggiungere le annotazioni necessarie al gestore di code.
    a. Fare clic sulla scheda **YAML**.
    b. Individuare il campo `spec` .
    c. Aggiungere le annotazioni richieste al gestore di code. Creare il campo `annotations` se non è già presente.
    d. Fare clic su **Salva**.

Il pod Queue Manager si riavvia e le annotazioni vengono applicate.

c) Ripetere questi passaggi per altri gestori di code, se necessario.

- **Opzione 2:** Aggiungere annotazioni utilizzando l'interfaccia a riga di comando (CLI) di OpenShift .

a) Accedere alla CLI di Red Hat OpenShift utilizzando il comando `oc login` .

b) Impostate il vostro `project` (spazio dei nomi) su quello in cui è distribuito il vostro gestore di code:

```
oc project <QUEUE-MANAGER-NAMESPACE>
```

c) Identificare il nome del gestore di code che si desidera aggiornare:

```
oc get queuemanager
```

d) Aggiungere le annotazioni richieste al gestore di code modificando il gestore di code:

```
oc edit queuemanager <QUEUE-MANAGER-NAME>
```

e) Ripetere questi passaggi per altri gestori di code, se necessario.

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.

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 Operator2.1.0 Disabilitazione degli aggiornamenti dei valori predefiniti per la specifica del gestore code

IBM MQ Operator aggiorna tutti i valori non specificati nella specifica del gestore code con i relativi valori predefiniti. È possibile disabilitare questo comportamento se si desidera evitare qualsiasi modifica alla specifica del gestore code. I campi di stato del gestore code sono ancora aggiornati.

Procedura

- Disabilita gli aggiornamenti dei valori predefiniti del gestore code.

  Aggiungere la seguente annotazione in `metadata`. Ad esempio:

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: quickstart-cp4i
  annotations:
    "com.ibm.mq/write-defaults-spec" : "false"
```

  Nota: Gli esempi quickstart hanno questa annotazione applicata per impostazione predefinita.

Esecuzione del contenitore IBM MQ con un file system root di sola lettura

È possibile configurare il contenitore IBM MQ per l'esecuzione con un file system root di sola lettura. Ciò impedisce agli aggressori di copiare ed eseguire codice doloso nel contenitore.

Informazioni su questa attività

L'abilitazione del file system root di sola lettura rende immutabili i file del contenitore. Nel file system del contenitore, i file possono essere visualizzati ma non modificati e non possono essere creati nuovi file. I file possono essere modificati o creati solo su un sistema di file montato.

Quando è abilitato un file system root di sola lettura, vengono creati due volumi effimeri `Scratch` e `Tmp` e montati rispettivamente nelle directory `/run` e `/tmp` nel contenitore.

- Il volume `Svuotato` contiene i file, i keystore e altri file utilizzati per la configurazione del gestore code.
- Il volume `Tmp` contiene i file diagnostici, ad esempio i file RAS del gestore code.

Poiché questi volumi sono effimeri, i file su questi volumi vengono persi al riavvio del pod.

Il tipo di volume creato per i dati del gestore code dipende dal tipo di memoria. Per impostazione predefinita, viene montato un volume persistente. Oppure, se il tipo di memoria è `effimero`, viene montato un volume effimero. Se la dimensione dei dati nel volume supera il valore specificato per la proprietà **`sizeLimit`** , Kubernetes può espellere il contenitore e crearne uno nuovo.

Un file system root di sola lettura non è abilitato per impostazione predefinita. Per abilitarla, completare la seguente procedura:

Procedura

1. Utilizzare l'API `spec.securityContext` per abilitare il file system root di sola lettura.

   Per il gestore code, impostare la proprietà **`readOnlyRootFilesystem`** in “.spec.securityContext” a pagina 269 su `true`.

   IBM MQ Operator crea due volumi effimeri, `Scratch` e `Tmp`.
2. Opzionale: Impostare o modificare il tipo di memoria dei dati del gestore code.

   Per impostazione predefinita, una richiesta di volume persistente è montata in `/mnt/mqm`. In alternativa, se la proprietà **`type`** è impostata su `effimero` in “.spec.queueManager.storage.queueManager” a pagina 267, viene creato e montato un volume effimero.
3. Per ogni volume effimero, considerare attentamente la crescita dei dati. Impostare di conseguenza il valore della proprietà **`sizeLimit`** , incluse le unità SI.
   - Per il volume temporaneo `Scratch` , impostare la proprietà **`sizeLimit`** in “.spec.queueManager.storage.scratch” a pagina 268. Il valore predefinito è "100M".
   - Per il volume temporaneo `Tmp` , impostare la proprietà **`sizeLimit`** in “.spec.queueManager.storage.tmp” a pagina 268. Il valore predefinito è "2Gi".
   - Se il valore **`type`** del volume del gestore code è impostato su `effimero`, impostare la proprietà **`sizeLimit`** in “.spec.queueManager.storage.queueManager” a pagina 267. Il valore predefinito è "2Gi".

OpenShift V 9.4.0 Configurazione di IBM MQ Console con un registro di base utilizzando IBM MQ Operator

Per accedere a IBM MQ Console, è possibile fornire la propria configurazione al gestore code.

Prima di iniziare

Se si sta distribuendo un gestore code con una licenza IBM MQ Advanced for Developers , è presente una semplice configurazione integrata. Consultare “YAML del gestore code di esempio che descrive come specificare le password per gli utenti admin e app” a pagina 293. Se si sta distribuendo un gestore code di licenze IBM Cloud Pak for Integration , è possibile abilitare l'integrazione con IBM Cloud Pak for Integration Keycloak per accedere a IBM MQ Console utilizzando SSO (Single Sign - On). Vedere “Connessione al IBM MQ Console distribuito in un cluster Red Hat OpenShift” a pagina 217.

Procedura

1. **Creare una password e codificarla utilizzando `securityUtility`.**

   Un `ConfigMap` viene utilizzato per memorizzare le credenziali che si utilizzano per accedere al proprio gestore code. Per migliorare la sicurezza, codificare queste credenziali con il comando securityUtility.

   In alternativa, puoi utilizzare un segreto, che protegge le credenziali nel livello Kubernetes . Tuttavia, gli strumenti di monitoraggio o di risoluzione dei problemi potrebbero esporre il file sottostante in modo non sicuro.
2. Opzionale: **Accedi alla CLI (command line interface) Red Hat OpenShift .**

   Se utilizzi la CLI OpenShift , accedi utilizzando `oc login`.

In alternativa, puoi utilizzare la console OpenShift .

3. **Crea un `ConfigMap` con la tua configurazione.**

   Per assistenza nella creazione della configurazione XML, consultare IBM MQ Console e REST API security.

   Il seguente esempio crea un utente all'interno del gruppo `MQWebAdminGroup`. Ai componenti di `MQWebAdminGroup` viene assegnato il ruolo `MQWebAdmin` . In questo esempio:

   - **È necessario** sostituire *USERNAME* e *PASSWORD* con i propri valori. Notare che *USERNAME* viene utilizzato due volte nell'esempio.

     È necessario specificare *NAMESPACE* come quello in cui IBM MQ Operator è distribuito e dove il gestore code sarà o già è distribuito.

   a) Utilizza la console OpenShift o la riga di comando per creare il seguente `ConfigMap`:

   ```
   kind: ConfigMap
   apiVersion: v1
   metadata:
     name: mqwebuserconfigmap
     namespace: NAMESPACE
   data:
     mqwebuser.xml: |
       <?xml version="1.0" encoding="UTF-8"?>
       <server>
         <featureManager>
           <feature>appSecurity-2.0</feature>
           <feature>basicAuthenticationMQ-1.0</feature>
         </featureManager>
         <enterpriseApplication id="com.ibm.mq.console">
           <application-bnd>
             <security-role name="MQWebAdmin">
               <group name="MQWebAdminGroup" realm="defaultRealm"/>
             </security-role>
           </application-bnd>
         </enterpriseApplication>
         <basicRegistry id="basic" realm="defaultRealm">
           <user name="USERNAME" password="PASSWORD"/>
           <group name="MQWebAdminGroup">
             <member name="USERNAME"/>
           </group>
         </basicRegistry>
         <sslDefault sslRef="mqDefaultSSLConfig"/>
       </server>
   ```

   b) Opzionale: Se si utilizza la riga comandi, applicare `ConfigMap`:

   ```
   oc apply -f mqwebuserconfigmap.yaml
   ```

   Per i passi rimanenti, scegliere una delle opzioni seguenti:

   - Distribuire un nuovo gestore code con la configurazione per accedere a IBM MQ Console.
   - Applicare la configurazione che fornisce l'accesso IBM MQ Console a un gestore code esistente.

4. Opzionale: **Distribuire un nuovo gestore code con la configurazione per accedere a IBM MQ Console**

   a) Creare il gestore code.

   Impostare i provider di autenticazione e autorizzazione su `manuale` e fornire il `ConfigMap` `mqwebuserconfigmap` appena creato mediante una delle seguenti opzioni:

   - Opzione 1: tramite il gestore code YAML

     Aggiungi il seguente codice nella sezione `web` dello YAML del gestore code:

     ```
     ...
     web:
         enabled: true
         console:
           authentication:
             provider: manual
           authorization:
             provider: manual
         manualConfig:
     ```

```
      configMap:
        name: mqwebuserconfigmap
```

- Opzione 2: tramite la vista modulo della console di OpenShift :
  i) Sulla console OpenShift , selezionare **Operatori** > **Operatori installati**.
  ii) Selezionare la distribuzione di IBM MQ Operator.
  iii) Selezionare **Gestore code** e fare clic su **Crea QueueManager**.
  iv) Selezionare le opzioni rilevanti per il gestore code.
  v) Selezionare **Web** e impostare **Abilita server Web** su `true`.
  vi) Aprire la casella di elenco **Configurazione avanzata** .
  vii) Nella casella di elenco **Console** , impostare il **provider** per **Authentication** e **Authorization** su `manual`.
  viii) Aprire la casella di elenco **Configurazione** .
  ix) Aprire la casella di elenco **ConfigMap** e selezionare ConfigMap `mqwebuserconfigmap` creato nel passo “3” a pagina 166.
  x) Fai clic su **Crea**.

  È ora possibile accedere al IBM MQ Console del nuovo gestore code tramite le credenziali specificate nel `ConfigMap` creato nel passo “3” a pagina 166.

5. Opzionale: **Applicare la configurazione che abilita IBM MQ Console per un gestore code esistente**

   Modificare lo YAML del gestore code per cui si sta abilitando IBM MQ Console:

   a. Sulla console di OpenShift selezionare **Operatori** > **Operatori installati**.
   b. Selezionare la distribuzione di IBM MQ Operator.
   c. Selezionare **Gestore code** e selezionare il nome del proprio gestore code.
   d. Selezionare **YAML**.
   e. Sostituire la sezione `web` esistente dello YAML del gestore code con il seguente codice:

```
...
web:
    enabled: true
    console:
      authentication:
        provider: manual
      authorization:
        provider: manual
    manualConfig:
      configMap:
        name: mqwebuserconfigmap
```

   f. Fare clic su **Salva**.

   Puoi ora accedere al IBM MQ Console del tuo gestore code esistente tramite le credenziali specificate nel `ConfigMap` creato nel passo “3” a pagina 166.

OpenShift V 9.4.0 V 9.4.0 Espansione dei volumi persistenti

Se il provider di memoria supporta l'espansione del volume, utilizzare questa attività per espandere un volume persistente. A seconda del provider di memoria, l'espansione potrebbe verificarsi in linea o non in linea.

Prima di iniziare

Una corretta espansione del volume si basa sul provider di memoria per soddisfare la richiesta di espansione. Fare riferimento alla documentazione dei provider di memoria per determinare se il ridimensionamento in linea è supportato e per informazioni sulle procedure di ridimensionamento non in linea.

Se il provider di memoria non è in grado di soddisfare la richiesta di espansione, la richiesta di volume persistente potrebbe entrare in uno stato con avvertenze o errori. Se l'espansione non riesce, un amministratore OpenShift può ripristinare manualmente lo stato Reclamo volume persistente e annullare l'espansione. Consultare Ripristino da un errore durante l'espansione dei volumi nella documentazione Red Hat OpenShift .

Informazioni su questa attività

Per aiutare con la gestione dell'archiviazione persistente, Kubernetes definisce due risorse API:

- Un PersistentVolume (PV), che è una parte di archiviazione nel cluster di cui è stato eseguito il provisioning da un amministratore o di cui è stato eseguito il provisioning in modo dinamico utilizzando le classi di archiviazione. È possibile eseguire il provisioning in modo statico o dinamico.
- Un'attestazione PersistentVolume(PVC), che è una richiesta di archiviazione da parte di un utente. Funge anche da controllo di richiesta alla risorsa.

Per ulteriori informazioni, vedi Volumi persistenti nella documentazione di Kubernetes .

Avvertenza:

- Se la classe di memoria utilizzata per creare le PVC del gestore code non supporta il ridimensionamento online, si verifica il ridimensionamento offline. Durante il ridimensionamento offline l'intervento dell'utente è necessario per completare l'espansione del volume, in modo che i gestori code riscontrino un tempo di inattività.
- Per il ridimensionamento offline dei volumi condivisi per i gestori code a più istanze, sia i pod attivi che quelli in standby devono essere disattivati contemporaneamente quando si esegue l'intervento dell'utente.
- OpenShift non supporta la riduzione della dimensione delle PVC. Se si tenta di ridurre la dimensione dei volumi persistenti, il gestore code verrà impostato sullo stato 'Non riuscito'.
- Questa procedura non si applica ai volumi effimeri.

Per espandere un PV utilizzato dal contenitore IBM MQ , completa la seguente procedura.

Procedura

1. **Preparazione all'espansione dei volumi**
   a) Decidere quali volumi espandere.
   b) Determinare la classe o le classi di archiviazione utilizzate dai propri volumi.
   Ad esempio:

```
spec:
  queueManager:
    storage:
      persistedData:
        enabled: true
        type: persistent-claim
        class: ocs-storagecluster-cephfs (1)
      queueManager:
        type: persistent-claim
      recoveryLogs:
        enabled: true
        type: persistent-claim
      defaultClass: ocs-storagecluster-ceph-rbd (2)
```

   Note:

   - (1) Se il volume definisce una classe di memoria specifica, viene utilizzata dalle PVC di questo tipo.
   - (2) Se è impostato **defaultClass** , questa classe di memoria viene utilizzata per tutti i volumi senza una classe di memoria specifica. Se **defaultClass** non è impostato e un tipo di volume non ha specificato una classe, viene utilizzata la classe di archiviazione predefinita per il cluster.

Puoi inoltre confermare la classe di archiviazione in uso descrivendo le PVC sottostanti. Ad esempio:

```
oc describe pvc pvc-name
```

c) Verificare che la classe di memoria supporti l'espansione del volume.

Una classe di archiviazione potrebbe avere la proprietà **.allowVolumeExpansion** definita:

- Se questa proprietà è impostata su `true`, l'espansione del volume è supportata.
- Se questa proprietà è impostata su `false`o non è definita, la classe di archiviazione non consente l'espansione del volume. In questo caso, fare riferimento alla documentazione del provider di memoria per verificare se questa funzione può essere abilitata.

È inoltre possibile descrivere una classe di memoria per determinare se supporta l'espansione del volume. Ad esempio:

```
oc describe sc storage-class-name
```

d) Fare riferimento alla documentazione del provider di memoria per verificare se viene utilizzata una procedura online o offline per l'espansione del volume.

Una procedura offline richiede che i pod del gestore code vengano riavviati manualmente, mentre una procedura online non lo fa. Fare riferimento alla documentazione del provider di memoria per le procedure di ridimensionamento non in linea.

e) Verificare se il gestore code ha una condizione di stato con il motivo 'StorageMismatch'.

Se il gestore code ha questa condizione di stato, i volumi elencati nella condizione vengono espansi se si abilita l'espansione del volume. Se non si desidera che ciò accada, modificare i campi della dimensione associati a ciascun tipo di volume nella definizione del gestore code in modo che corrispondano alle PVC di cui è stato eseguito il provisioning. La condizione di stato viene rimossa quando questa operazione viene eseguita per tutti i volumi non corrispondenti.

2. **Espandi volumi**

Avvertenza:

- Se è stato precedentemente modificato uno dei campi della dimensione del volume nella definizione del gestore code, i volumi iniziano ad espandersi quando **.allowVolumeExpansion** è impostato su `true` nella definizione del gestore code.
- Il provider di memoria potrebbe avere limitazioni sulla dimensione massima di un volume a causa delle limitazioni del file system o della disponibilità dell'hardware locale. Per evitare errori, convalidare queste limitazioni nella documentazione del provider di memoria prima di espandere i volumi.
- Le riduzioni della dimensione del PVC non sono supportate da OpenShift. Se si espande la dimensione di un volume, non è possibile ridurla. Se il tuo tentativo di eseguire questa operazione non riesce, IBM MQ Operator non può riportare la PVC al suo stato originale.

Esempio di definizione del gestore code che illustra l'espansione del volume:

```
spec:
  queueManager:
    storage:
      allowVolumeExpansion: true (A)
      persistedData:
        enabled: true
        type: persistent-claim
        size: 3Gi (B)
      queueManager:
        type: persistent-claim
        size: 4Gi (B)
      recoveryLogs:
        enabled: true
        type: persistent-claim
        size: 3Gi (B)
```

a) Per consentire l'espansione del volume per il gestore code, impostare il campo **.spec.queueManager.storage.allowVolumeExpansion** (A) sul proprio gestore code su `true`.

b) È ora possibile aumentare i campi di dimensione (B) per qualsiasi tipo di volume abilitato. L'applicazione di queste modifiche avvierà l'espansione del volume.

3. **Convalidare che le PVC sono state ridimensionate.**

   Note:

   - L'espansione del volume può richiedere del tempo. Se la convalida non ha esito positivo, la prima volta considerare l'attesa di alcuni minuti e la convalida di nuovo.
   - L'espansione del volume viene completata solo senza l'azione dell'utente quando viene eseguito un ridimensionamento in linea.
   - Alcuni provider di memoria completano la dimensione di memoria richiesta. Il volume espanso deve avere la stessa dimensione o una dimensione maggiore della richiesta.

   a) Controllare le condizioni di stato del proprio gestore code. Fare riferimento alla seguente tabella per le condizioni, le spiegazioni e le azioni suggerite.

Tabella 1. Condizioni di stato per l'archiviazione

CONDITION	**MESSAGGIO**	**Spiegazione**
`StorageMismatch`	`Storage sizes defined in the QueueManager resource do not match the capacity of one or more provisioned PVCs [pvc-list]. AllowVolumeExpansion is set to false in the QueueManager resource so the MQ Operator will not attempt to reconcile these differences.`	L'espansione del volume non si verifica perché **.allowVolumeExpansion** non è stato impostato su `true` nella definizione del gestore code.
`StorageExpansionPending`	`Volume expansion is pending for the following PVCs [pvc-list]`	L'espansione del volume è ancora in corso. Se questa condizione di stato persiste per un periodo di tempo prolungato, seguire la procedura riportata di seguito per raccogliere ulteriori informazioni perché potrebbe verificarsi un ridimensionamento non in linea o un errore di ridimensionamento.

Tabella 1. Condizioni di stato per l'archiviazione (Continua)		
CONDITION	**MESSAGGIO**	**Spiegazione**
`Failed`	Ci sono molti possibili messaggi relativi alla memoria che possono creare una condizione di stato `'Failed'` . Ad esempio: `'MQ Queue Manager failed to deploy: persistentvolumeclaims "<pvc>" is forbidden: only dynamically provisioned pvc can be resized and the storageclass the provisions the pvc must support resize.'`	Se il gestore code presenta condizioni di stato `'Failed'` con testo che fa riferimento alla memoria, fare riferimento al messaggio all'interno della condizione di stato. Il messaggio di esempio fornito qui è causato dall'utilizzo di una classe di memoria che non supporta l'espansione.

b) Per ogni PVC espanso, verificare che la capacità sia aumentata in modo che corrisponda o sia maggiore del valore specificato nella definizione del gestore code.

I gestori code HA potrebbero avere più PVC di ogni tipo. Per ottenere la capacità di una PVC, immetti questo comando:

```
oc get pvc pvc-name -o template --template '{{.status.capacity.storage}}'
```

c) Controlla che la PVC non abbia condizioni di stato o eventi che suggeriscono un ridimensionamento non riuscito:

```
oc describe pvc pvc-name
```

- La tua PVC potrebbe avere la condizione di stato `FileSystemResizePending` con il messaggio 'Waiting for user to (re-) start a pod to finish file system resize of volume on node'. Questa condizione di stato viene generata per i ridimensionamenti in linea e non in linea. Per un ridimensionamento online, questa condizione di stato scompare senza l'azione dell'utente una volta completato il ridimensionamento online.
- Se la tua PVC ha una condizione di evento o di stato che indica un ridimensionamento non riuscito, vedi Ripristino da un errore durante l'espansione dei volumi nella documentazione di Red Hat OpenShift .

d) Verificare che i pod del gestore code non abbiano condizioni di stato o eventi che suggeriscono un ridimensionamento non riuscito. Per le distribuzioni HA, controllare ogni replica.

```
oc describe pod queue-manager-pod-name
```

- Se il tuo pod ha un evento o una condizione di stato che indica un ridimensionamento non riuscito, vedi Ripristino da un errore durante l'espansione dei volumi nella documentazione di Red Hat OpenShift . Il testo dell'errore potrebbe aiutare a risolvere il problema o impedire che si verifichi lo stesso problema se si tenta di ridimensionare dopo il ripristino.

4. **Riavviare i pod quando si ridimensiona offline**

Se il provider di memoria utilizza una procedura di ridimensionamento non in linea durante l'espansione dei volumi, per completare l'espansione del volume è necessario riavviare i pod del gestore code che montano i volumi ridimensionati.

Per i gestori code a più istanze, i log di ripristino e i volumi di dati persistenti sono condivisi tra i pod attivi e in standby. Per il completamento del ridimensionamento di questi volumi, ridurre entrambi i pod contemporaneamente.

Fare riferimento alla documentazione del provider di memoria per la procedura di ridimensionamento offline.

OpenShift V 9.4.1 Correzione delle annotazioni di licenza per i gestori di code distribuiti

Prima della versione 9.4.1.0-r1 del gestore di code 'IBM MQ, alcuni gestori di code venivano distribuiti con annotazioni di licenza errate. Dalla versione 9.4.1.0-r1 tutti i gestori di code vengono distribuiti automaticamente con le annotazioni di licenza corrette.

Informazioni su questa attività

Per i gestori di code 'IBM MQ già distribuiti ci sono due opzioni per correggere le annotazioni di licenza:

- Aggiornare il gestore di code alla versione 9.4.1.0-r1 (richiede 'IBM MQ Operator 3 3.3.0 o più recente). Vedere “Aggiornamento di IBM MQ Operator e dei gestori code” a pagina 69. Si noti che se lo YAML del gestore di code contiene annotazioni di licenza non valide, è necessario risolverle come richiesto durante l'aggiornamento di ciascun gestore di code.
- Per i vecchi gestori di code che non sono stati aggiornati e che sono distribuiti utilizzando i seguenti usi della licenza, le annotazioni della licenza possono essere corrette aggiornando lo YAML del gestore di code. Completare i passaggi seguenti per aggiungere le annotazioni specificate allo YAML del gestore di code. Si noti che questo processo deve essere completato individualmente per ogni gestore di code.

  Importante: L'aggiornamento dello YAML del gestore delle code provoca il riavvio del pod.

Utilizzo della licenza interessata:

- IBM MQ Advanced for Non-Production Environment con diritto al CP4I
- IBM MQ Advanced for Developers
- IBM MQ Advanced Container Multi Instance
- IBM MQ Advanced Container Multi Instance con diritto al CP4I
- IBM MQ Advanced Container Multi Instance for Non-Production Environment con diritto al CP4I

Procedura

Aggiornare le annotazioni di licenza dei gestori di code esistenti seguendo le istruzioni riportate in “Aggiungere annotazioni ai gestori di code utilizzando l'opzione IBM MQ Operator” a pagina 162, utilizzando le annotazioni di licenza richieste per ciascun gestore di code, in base alla propria abilitazione.

Annotazioni richieste per la licenza

IBM MQ Advanced for Non-Production Environment con diritto al CP4I

```
spec:
  annotations:
    productName: IBM MQ Advanced for Non-Production Environment
```

IBM MQ Advanced for Developers

```
spec:
  annotations:
    productName: IBM MQ Advanced for Developers (Non-Warranted)
    productChargedContainers: [container_name]
```

dove *nome_contenitore* si trova nel pod YAML del gestore di code, specificato come 'spec.containers[0].name

IBM MQ Advanced Container Multi Instance

```
spec:
  annotations:
```

```
    productID: bd35bff411bb47c2a3f3a4590f33a8ef
    productName: IBM MQ Advanced Container Multi Instance
```

IBM MQ Advanced Container Multi Instance con diritto al CP4I

```
spec:
  annotations:
    productID: bd35bff411bb47c2a3f3a4590f33a8ef
    productName: IBM MQ Advanced Container Multi Instance
    productCloudpakRatio: 5:3
```

IBM MQ Advanced Container Multi Instance for Non-Production Environment con diritto al CP4I

```
spec:
  annotations:
    productID: 31f844f7a96b49749130cd0708fdbb17
    productName: IBM MQ Advanced Container Multi Instance for Non-Production Environments
    productCloudpakRatio: 10:3
```

Kubernetes Distribuzione e configurazione dei gestori di code su Kubernetes

Come distribuire i gestori di code su Kubernetes utilizzando risorse native.

Informazioni su questa attività

V 9.4.1 MQ Adv. Kubernetes Esempio: Configurazione di un semplice gestore di code in 'Kubernetes

Questo esempio implementa un semplice gestore di code "quick start" che utilizza uno storage effimero (non persistente) e disattiva la sicurezza di IBM MQ. I messaggi non vengono mantenuti tra i riavvii del gestore di code. È possibile modificare la configurazione per cambiare molte impostazioni del gestore di code.

Prima di iniziare

Per completare questo esempio, è necessario aver completato i seguenti prerequisiti:

- Creare uno spazio dei nomi 'Kubernetes per questo esempio.
- Alla riga di comando, accedere al cluster 'Kubernetes e passare allo spazio dei nomi di cui sopra.
- Assicuratevi di aver seguito i passi di '“Preparare l'uso di 'Kubernetes creando un segreto di pull” a pagina 86 nel namespace di cui sopra.

Informazioni su questa attività

Questo esempio fornisce 'Service, 'ServiceAccount e 'StatefulSet YAML per definire un gestore di code da distribuire in 'Kubernetes. Questo esempio richiede un minimo di 'IBM MQ Advanced 9.4.1.

Procedura

1. Creare un " Service per fornire un indirizzo IP coerente per il traffico " IBM MQ.

   Assicurarsi di essere nello spazio dei nomi creato in precedenza (vedere Prima di iniziare), quindi applicare il seguente YAML usando la riga di comando 'Kubernetes o la console.

   ```
   apiVersion: v1
   kind: Service
   metadata:
     labels:
       app.kubernetes.io/component: integration
       app.kubernetes.io/instance: quickstart
       app.kubernetes.io/name: ibm-mq
       app.kubernetes.io/version: 9.4.3.0
     name: quickstart-ibm-mq
   ```

```
spec:
  ports:
  - name: qmgr
    port: 1414
    protocol: TCP
    targetPort: 1414
  - name: metrics
    port: 9157
    protocol: TCP
    targetPort: 9157
  selector:
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: quickstart
    app.kubernetes.io/name: ibm-mq
  type: ClusterIP
```

2. Creare un 'ServiceAccount da utilizzare per il gestore delle code.

   L'esempio " ServiceAccount di seguito riportato si riferisce a un " imagePullSecret che viene utilizzato per un controllo dei diritti quando l'immagine viene estratta. Per ulteriori informazioni, vedere " “Preparare l'uso di 'Kubernetes creando un segreto di pull” a pagina 86.

```
kind: ServiceAccount
apiVersion: v1
metadata:
  name: quickstart-ibm-mq
  labels:
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: quickstart
    app.kubernetes.io/name: ibm-mq
    app.kubernetes.io/version: 9.4.3.0
imagePullSecrets:
  - name: ibm-entitlement-key
```

3. Creare il gestore di code Pod, gestito da un 'StatefulSet.

   Nel seguente YAML, il nome 'namespace è assunto come 'mq-test. Modificatelo in modo che corrisponda al nome che state usando nell'ambiente 'Kubernetes per questo esempio.

   Questo YAML crea una singola istanza di gestore di code nell'ambiente 'Kubernetes nello spazio dei nomi specificato. Non viene creato alcun volume di persistenza per questa istanza, perché il nostro esempio è per l'archiviazione effimera.

   Nota: Questo esempio utilizza le annotazioni di IBM MQ Advanced . Per utilizzare una licenza IBM MQ , modificare le annotazioni in spec.template.metadata con i valori specificati in IBM MQ o IBM MQ for Non-Production Environment in “IBM MQ annotazioni sulla licenza del contenitore” a pagina 281.

```
kind: StatefulSet
apiVersion: apps/v1
metadata:
  name: quickstart-ibm-mq
  namespace: mq-test
  labels:
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: quickstart
    app.kubernetes.io/name: ibm-mq
    app.kubernetes.io/version: 9.4.3.0
spec:
  persistentVolumeClaimRetentionPolicy:
    whenDeleted: Retain
    whenScaled: Retain
  replicas: 1
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app.kubernetes.io/component: integration
      app.kubernetes.io/instance: quickstart
      app.kubernetes.io/name: ibm-mq
  template:
    metadata:
      labels:
        app.kubernetes.io/component: integration
        app.kubernetes.io/instance: quickstart
        app.kubernetes.io/name: ibm-mq
        app.kubernetes.io/version: 9.4.3.0
        statefulSetName: quickstart-ibm-mq
```

```
      annotations:
# These annotations are critical for the IBM License Service to charge you correctly
        productChargedContainers: qmgr
        productID: 208423bb063c43288328b1d788745b0c
        productMetric: PROCESSOR_VALUE_UNIT
        productName: IBM MQ Advanced
        productVersion: 9.4.3.0
    spec:
      restartPolicy: Always
      serviceAccountName: quickstart-ibm-mq
      terminationGracePeriodSeconds: 30
      securityContext: {}
      containers:
        - resources:
            limits:
              cpu: '1'
              memory: 1Gi
            requests:
              cpu: '1'
              memory: 1Gi
          readinessProbe:
            exec:
              command:
                - chkmqready
            initialDelaySeconds: 10
            timeoutSeconds: 3
            periodSeconds: 5
            successThreshold: 1
            failureThreshold: 1
          terminationMessagePath: /run/termination-log
          name: qmgr
          livenessProbe:
            exec:
              command:
                - chkmqhealthy
            initialDelaySeconds: 90
            timeoutSeconds: 5
            periodSeconds: 10
            successThreshold: 1
            failureThreshold: 1
          env:
            - name: LICENSE
              # if you accept the license, change the following value to "accept"
              value: view
            - name: MQ_QMGR_NAME
              value: QUICKSTART
            - name: MQ_MULTI_INSTANCE
              value: 'false'
            - name: MQ_ENABLE_METRICS
              value: 'true'
            - name: MQ_ENABLE_EMBEDDED_WEB_SERVER
              value: 'false'
            - name: MQ_LOGGING_CONSOLE_FORMAT
              value: basic
            - name: DEBUG
              value: 'false'
            - name: MQ_ENABLE_TRACE_STRMQM
              value: 'false'
            - name: MQ_EPHEMERAL_PREFIX
              value: /run/mqm
            - name: MQ_GRACE_PERIOD
              value: '29'
            - name: MQ_NATIVE_HA
              value: 'false'
          securityContext:
            allowPrivilegeEscalation: false
            capabilities:
              drop:
                - ALL
            privileged: false
            runAsNonRoot: true
            readOnlyRootFilesystem: false
          ports:
            - containerPort: 1414
              protocol: TCP
            - containerPort: 9157
              protocol: TCP
          terminationMessagePolicy: File
# The image to pull.  This example uses a prebuilt MQ Advanced image from IBM Container
Registry
          image: cp.icr.io/cp/ibm-mqadvanced-server:9.4.1.0-r1
          imagePullPolicy: IfNotPresent
```

```
      serviceAccount: quickstart-ibm-mq
      dnsPolicy: ClusterFirst
  serviceName: qm
  podManagementPolicy: OrderedReady
  updateStrategy:
    type: OnDelete
```

4. Confermare che il gestore di code è in esecuzione

   Il gestore di code viene ora distribuito. Verificare che sia in stato 'Running, ad esempio elencando i pod del gestore delle code:

   ```
   kubectl get pods --selector app.kubernetes.io/instance=quickstart
   ```

   Si noti che questo deve restituire un singolo pod e che il campo 'READY deve mostrare il valore '1/1.

5. Visualizzare lo stato del gestore delle code

   Eseguite il seguente comando, specificando il nome completo del pod del gestore di code:

   ```
   kubectl exec -t <qmgr_pod_name> -- dspmq -m QUICKSTART
   ```

   La risposta dovrebbe essere simile alla seguente:

   ```
   QMNAME(QUICKSTART) STATUS(Running)
   ```

Risultati

Avete distribuito con successo un semplice gestore di code 'IBM MQ, di tipo istanza singola, sulla vostra piattaforma 'Kubernetes.

V 9.4.1 MQ Adv. Kubernetes Esempio: Configurazione dell'HA nativo in 'Kubernetes

Questo esempio distribuisce un gestore di code utilizzando la funzione nativa di alta disponibilità in 'Kubernetes, usando direttamente le risorse YAML. Mutual TLS è usato per l'autenticazione, per mappare da un certificato TLS a un'identità nel gestore di code.

Prima di iniziare

Nota: Questo esempio utilizza una licenza IBM MQ Advanced . Da IBM MQ 9.4.2, è anche possibile configurare i gestori di code con licenza IBM MQ per utilizzare le funzionalità Native HA e Cross-Region Replication aggiungendo licenze aggiuntive a un gestore di code. Per ulteriori informazioni, fare riferimento a “Configurazione di Native HA e Cross-Region Replication su gestori di code con licenza IBM MQ in Kubernetes” a pagina 186.

Per completare questo esempio, è necessario aver completato i seguenti prerequisiti:

- Creare uno spazio dei nomi 'Kubernetes per questo esempio.
- Alla riga di comando, accedere al cluster 'Kubernetes e passare allo spazio dei nomi di cui sopra.
- Assicuratevi di aver seguito i passi di '“Preparare l'uso di 'Kubernetes creando un segreto di pull” a pagina 86 nel namespace di cui sopra.

Informazioni su questa attività

Questo esempio fornisce 'Service, 'ServiceAccount e 'StatefulSet YAML per definire un gestore di code da distribuire in 'Kubernetes. Vengono inoltre descritti i passaggi aggiuntivi necessari per distribuire il gestore di code con TLS abilitato. Questo esempio richiede un minimo di 'IBM MQ Advanced 9.4.1.

Procedura

1. Creare una coppia di certificati come descritto in '“Creazione di una PKI autofirmata utilizzando OpenSSL” a pagina 104.
2. Creare una mappa di configurazione contenente i comandi MQSC e un file INI.

   Creare un 'Kubernetes 'ConfigMap contenente i comandi MQSC per creare una nuova coda e un canale 'SVRCONN e per aggiungere un record di autenticazione del canale che consenta l'accesso al canale.

   Assicurarsi di essere nello spazio dei nomi creato in precedenza (vedere Prima di iniziare), quindi applicare il seguente YAML usando la riga di comando 'Kubernetes.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: example-nativeha-configmap
data:
  example.mqsc: |
    DEFINE CHANNEL('MTLS.SVRCONN') CHLTYPE(SVRCONN) SSLCAUTH(REQUIRED)
SSLCIPH('ANY_TLS13_OR_HIGHER') REPLACE
    SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=*') USERSRC(NOACCESS)
ACTION(REPLACE)
    SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=example-app1') USERSRC(MAP)
MCAUSER('app1') ACTION(REPLACE)
    SET AUTHREC PRINCIPAL('app1') OBJTYPE(QMGR) AUTHADD(CONNECT,INQ)
    DEFINE QLOCAL('EXAMPLE.QUEUE') REPLACE
    SET AUTHREC PROFILE('EXAMPLE.QUEUE') PRINCIPAL('app1') OBJTYPE(QUEUE)
AUTHADD(BROWSE,PUT,GET,INQ)
  example.ini: |
    Service:
        Name=AuthorizationService
        EntryPoints=14
        SecurityPolicy=UserExternal
    NativeHAInstance:
      Name=exampleqm-ibm-mq-0
      ReplicationAddress=exampleqm-ibm-mq-replica-0(9414)
    NativeHAInstance:
      Name=exampleqm-ibm-mq-1
      ReplicationAddress=exampleqm-ibm-mq-replica-1(9414)
    NativeHAInstance:
      Name=exampleqm-ibm-mq-2
      ReplicationAddress=exampleqm-ibm-mq-replica-2(9414)
```

   L'MQSC definisce un canale chiamato *MTLS.SVRCONN* e una coda chiamata *EXAMPLE.QUEUE*. Il canale è configurato in modo da consentire l'accesso solo ai client che presentano un certificato con un "nome comune" di *example-app1*. Si tratta del nome comune utilizzato in uno dei certificati creati nel passaggio " “1” a pagina 177. Le connessioni su questo canale con questo nome comune sono mappate su un ID utente di *app1*, che è autorizzato a connettersi al gestore di code e ad accedere alla coda di esempio. Il file INI abilita un criterio di sicurezza che significa che l'ID utente *app1* non deve esistere in un registro utenti esterno: esiste solo come nome in questa configurazione. Configura anche gli indirizzi di replica da usare con Native HA.
3. Creare un " Service per fornire un indirizzo IP coerente per il traffico " IBM MQ.

```
apiVersion: v1
kind: Service
metadata:
  labels:
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: exampleqm
    app.kubernetes.io/name: ibm-mq
    app.kubernetes.io/version: 9.4.3.0
  name: exampleqm-ibm-mq
spec:
  ports:
  - name: qmgr
    port: 1414
    protocol: TCP
    targetPort: 1414
  - name: metrics
    port: 9157
    protocol: TCP
    targetPort: 9157
  selector:
```

```
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: exampleqm
    app.kubernetes.io/name: ibm-mq
  type: ClusterIP
```

4. Creare un " `Service` interno per ciascuna delle repliche Native HA da utilizzare per comunicare tra loro.

```
---
apiVersion: v1
kind: Service
metadata:
  labels:
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: exampleqm
    app.kubernetes.io/name: ibm-mq
    app.kubernetes.io/version: 9.4.3.0
  name: exampleqm-ibm-mq-replica-0
spec:
  ports:
  - port: 9414
    protocol: TCP
    targetPort: 9414
  publishNotReadyAddresses: true
  selector:
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: exampleqm
    app.kubernetes.io/name: ibm-mq
    statefulset.kubernetes.io/pod-name: exampleqm-ibm-mq-0
  type: ClusterIP
---
apiVersion: v1
kind: Service
metadata:
  labels:
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: exampleqm
    app.kubernetes.io/name: ibm-mq
    app.kubernetes.io/version: 9.4.3.0
  name: exampleqm-ibm-mq-replica-1
spec:
  ports:
  - port: 9414
    protocol: TCP
    targetPort: 9414
  publishNotReadyAddresses: true
  selector:
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: exampleqm
    app.kubernetes.io/name: ibm-mq
    statefulset.kubernetes.io/pod-name: exampleqm-ibm-mq-1
  type: ClusterIP
---
apiVersion: v1
kind: Service
metadata:
  labels:
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: exampleqm
    app.kubernetes.io/name: ibm-mq
    app.kubernetes.io/version: 9.4.3.0
  name: exampleqm-ibm-mq-replica-2
spec:
  ports:
  - port: 9414
    protocol: TCP
    targetPort: 9414
  publishNotReadyAddresses: true
  selector:
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: exampleqm
    app.kubernetes.io/name: ibm-mq
    statefulset.kubernetes.io/pod-name: exampleqm-ibm-mq-2
  type: ClusterIP
```

5. Creare un '`ServiceAccount` da utilizzare per il gestore delle code.

L'esempio " `ServiceAccount` di seguito riportato si riferisce a un " `imagePullSecret` che viene utilizzato per un controllo dei diritti quando l'immagine viene estratta. Per ulteriori informazioni, vedere " “Preparare l'uso di 'Kubernetes creando un segreto di pull” a pagina 86.

```
kind: ServiceAccount
apiVersion: v1
metadata:
  name: exampleqm-ibm-mq
  labels:
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: exampleqm
    app.kubernetes.io/name: ibm-mq
    app.kubernetes.io/version: 9.4.3.0
imagePullSecrets:
  - name: ibm-entitlement-key
```

6. Creare il gestore di code Pod, gestito da un '`StatefulSet`.

   Utilizzare il seguente YAML come esempio, apportando queste modifiche:

   - Cambiare il nome `namespace` da `mq-test` in modo che corrisponda al nome che si sta usando nell'ambiente Kubernetes per questo esempio.
   - Se si sta provando l'esempio su un cluster Kubernetes con meno di tre nodi disponibili, cancellare la sezione `affinity` .
   - Opzionalmente, modificare la licenza. L'esempio utilizza una licenza IBM MQ Advanced . Da IBM MQ 9.4.2, è anche possibile configurare i gestori di code con licenza IBM MQ per utilizzare le funzionalità Native HA e Cross-Region Replication, aggiungendo licenze aggiuntive a un gestore di code.

     a. Cambiate le annotazioni in `spec.template.metadata` con i valori specificati in IBM MQ o IBM MQ for Non-Production Environment in “IBM MQ annotazioni sulla licenza del contenitore” a pagina 281.

     b. Completare il passo “3” a pagina 187 in “Configurazione di Native HA e Cross-Region Replication su gestori di code con licenza IBM MQ in Kubernetes” a pagina 186. Questo passaggio aggiunge un CR `IBMLicensingDefinition` che aggiunge ulteriori annotazioni di licenza al gestore di code per le funzionalità Native HA e Cross-Region Replication

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  labels:
    app.kubernetes.io/component: integration
    app.kubernetes.io/instance: exampleqm
    app.kubernetes.io/name: ibm-mq
    app.kubernetes.io/version: 9.4.3.0
  name: exampleqm-ibm-mq
spec:
  persistentVolumeClaimRetentionPolicy:
    whenDeleted: Retain
    whenScaled: Retain
  updateStrategy:
    type: OnDelete
  podManagementPolicy: Parallel
  replicas: 3
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app.kubernetes.io/component: integration
      app.kubernetes.io/instance: exampleqm
      app.kubernetes.io/name: ibm-mq
  serviceName: qm
  template:
    metadata:
      # These annotations are critical for the IBM License Service to charge you correctly
      annotations:
        productChargedContainers: All
        productID: 208423bb063c43288328b1d788745b0c
        productMetric: PROCESSOR_VALUE_UNIT
        productName: IBM MQ Advanced
        productVersion: 9.4.3.0
      labels:
        app.kubernetes.io/component: integration
        app.kubernetes.io/instance: exampleqm
```

```
        app.kubernetes.io/name: ibm-mq
        app.kubernetes.io/version: 9.4.3.0
        statefulSetName: exampleqm-ibm-mq
    spec:
      # Create an anti-affinity rule, so the Native HA replicas don't end up on the same Node
      affinity:
        podAntiAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
          - labelSelector:
              matchExpressions:
              - key: statefulSetName
                operator: In
                values:
                - exampleqm-ibm-mq
            topologyKey: kubernetes.io/hostname
      containers:
      # Environment variables to supply to MQ container
      - env:
        - name: LICENSE
          # if you accept the license, change the following value to "accept"
          value: view
        - name: MQ_QMGR_NAME
          value: EXAMPLEQM
        - name: MQ_MULTI_INSTANCE
          value: "false"
        - name: MQ_ENABLE_METRICS
          value: "true"
        - name: MQ_ENABLE_EMBEDDED_WEB_SERVER
          value: "false"
        - name: MQ_LOGGING_CONSOLE_FORMAT
          value: basic
        - name: DEBUG
          value: "false"
        - name: MQ_ENABLE_TRACE_STRMQM
          value: "false"
        - name: MQ_EPHEMERAL_PREFIX
          value: /run/mqm
        - name: MQ_GRACE_PERIOD
          value: "29"
        - name: MQ_NATIVE_HA
          value: "true"
        # The image to pull.  This example uses a prebuilt MQ Advanced image from IBM
Container Registry
        image: cp.icr.io/cp/ibm-mqadvanced-server:9.4.3.0-r1
        imagePullPolicy: IfNotPresent
        livenessProbe:
          exec:
            command:
            - chkmqhealthy
          failureThreshold: 1
          periodSeconds: 10
          successThreshold: 1
          timeoutSeconds: 5
        name: qmgr
        ports:
        - containerPort: 1414
          protocol: TCP
        - containerPort: 9157
          protocol: TCP
        - containerPort: 9414
          protocol: TCP
        readinessProbe:
          exec:
            command:
            - chkmqready
          failureThreshold: 1
          periodSeconds: 5
          successThreshold: 1
          timeoutSeconds: 3
        # CPU and memory resources to allocate
        resources:
          limits:
            cpu: "1"
            memory: 1Gi
          requests:
            cpu: "1"
            memory: 1Gi
        # Container security context.  Disable all root capabilities and privileges.
        securityContext:
          allowPrivilegeEscalation: false
          capabilities:
            drop:
```

```
          - ALL
        privileged: false
        readOnlyRootFilesystem: true
        runAsNonRoot: true
      startupProbe:
        exec:
          command:
          - chkmqstarted
        failureThreshold: 24
        periodSeconds: 5
        successThreshold: 1
        timeoutSeconds: 5
      terminationMessagePath: /run/termination-log
      terminationMessagePolicy: File
      volumeMounts:
      # Mount the main queue manager persistent volume.  Sym-linked from /var/mqm
      - mountPath: /mnt/mqm
        name: data
      # Mount space for temporary files
      - mountPath: /run
        name: scratch
      - mountPath: /tmp
        name: tmp
      # Mount the "default" key for TLS
      - mountPath: /etc/mqm/pki/keys/default
        name: default
        readOnly: true
      # Mount the key to use for Native HA replication traffic
      - mountPath: /etc/mqm/ha/pki/keys/ha
        name: ha-tls
        readOnly: true
      # Mount MQSC to be applied when the queue manager starts
      - mountPath: /etc/mqm/example.mqsc
        name: cm-example-nativeha-configmap
        readOnly: true
        subPath: example.mqsc
      # Mount an INI file to be applied when the queue manager starts
      - mountPath: /etc/mqm/example.ini
        name: cm-example-nativeha-configmap
        readOnly: true
        subPath: example.ini
    restartPolicy: Always
    securityContext: {}
    serviceAccount: exampleqm-ibm-mq
    serviceAccountName: exampleqm-ibm-mq
    automountServiceAccountToken: false
    terminationGracePeriodSeconds: 30
    volumes:
    - emptyDir:
        sizeLimit: 100M
      name: scratch
    # Temporary directory is used by default to store runmqras output, so make it larger
    - emptyDir:
        sizeLimit: 2Gi
      name: tmp
    - name: default
      secret:
        defaultMode: 288
        items:
        - key: tls.key
          path: tls.key
        - key: tls.crt
          path: tls.crt
        - key: ca.crt
          path: ca.crt
        secretName: example-qm-tls
    - name: ha-tls
      secret:
        defaultMode: 288
        secretName: example-qm-tls
    - configMap:
        defaultMode: 420
        items:
        - key: example.mqsc
          path: example.mqsc
        - key: example.ini
          path: example.ini
        name: example-nativeha-configmap
      name: cm-example-nativeha-configmap
  volumeClaimTemplates:
  - apiVersion: v1
    kind: PersistentVolumeClaim
```

```
    metadata:
      labels:
        app.kubernetes.io/component: integration
        app.kubernetes.io/instance: exampleqm
        app.kubernetes.io/name: ibm-mq
      name: data
      namespace: mq-test
    spec:
      accessModes:
      - ReadWriteOnce
      resources:
        requests:
          storage: 2Gi
      volumeMode: Filesystem
```

Si noti che l'*esempio-qm-tls* " `Secret` è stato creato nel passo " “1” a pagina 177 e l'*esempio-nativeha-configmap* " `ConfigMap` è stato creato nel passo " “2” a pagina 177.

Per ogni istanza viene creato un volume persistente. Viene utilizzata la classe di archiviazione predefinita configurata nel cluster 'Kubernetes. Se non si dispone di una classe di archiviazione configurata come predefinita o si desidera utilizzare una classe di archiviazione diversa, aggiungere '`storageClassName: storage_class_name` sotto '`spec.volumeClaimTemplates[0].spec`.

I tre pod in un gestore di code Native HA replicano i dati sulla rete. Questo collegamento non è crittografato per impostazione predefinita, ma questo esempio utilizza il certificato del gestore di code per crittografare il traffico. È possibile specificare un certificato diverso per una maggiore sicurezza. Il Native HA TLS Secret deve essere un TLS Secret 'Kubernetes, che ha una struttura particolare (ad esempio, la chiave privata deve essere chiamata *tls.key*).

Si noti che il " `updateStrategy` è impostato su " `OnDelete`, il che influisce sul modo in cui gli aggiornamenti vengono distribuiti. Per ulteriori informazioni, fai riferimento a “Considerazioni sull'esecuzione di un aggiornamento continuo di un gestore code HA nativo” a pagina 44

7. Confermare che il gestore di code è in esecuzione.

   Il gestore di code viene ora distribuito. Prima di procedere, verificare che sia nello stato '`Running`, ad esempio elencando i pod del gestore delle code:

   ```
   kubectl get pods --selector app.kubernetes.io/instance=exampleqm
   ```

   L'ideale sarebbe collegarsi al gestore di code per verificare che sia in funzione. Tuttavia, la connessione al gestore delle code dall'esterno del cluster 'Kubernetes non rientra nell'ambito di queste istruzioni, perché può variare in modo significativo tra le distribuzioni 'Kubernetes e gli ambienti cloud.

8. Forzare il fallimento del pod attivo

   Per convalidare il ripristino automatico del gestore di code, simulare un guasto del pod:

   a) Visualizzare i pod attivi e in standby

      Eseguire nuovamente il comando precedente:

      ```
      kubectl get pods --selector app.kubernetes.io/instance=exampleqm
      ```

      Si noti che, nel campo " **READY**, il pod attivo restituisce il valore " `1/1`, mentre i pod di replica restituiscono il valore " `0/1`.

   b) Cancellare il pod attivo

      Eseguire il seguente comando, specificando il nome completo del pod attivo:

      ```
      kubectl delete pod exampleqm-ibm-mq-<active_pod_value>
      ```

   c) Visualizzare nuovamente lo stato del pod

      Esegui il seguente comando:

      ```
      kubectl get pods --selector app.kubernetes.io/instance=exampleqm
      ```

   d) Visualizzare lo stato del gestore delle code

      Eseguire il seguente comando, specificando il nome completo di uno degli altri pod:

```
kubectl exec -t <qmgr_pod_name> -- dspmq -o nativeha -x -m EXAMPLEQM
```

Lo stato mostra che l'istanza attiva è cambiata, ad esempio:

```
QMNAME(EXAMPLEQM) 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)
```

Risultati

È stato implementato con successo un gestore di code con alta disponibilità nativa e autenticazione TLS reciproca e si è verificato che si ripristina automaticamente quando il Pod attivo si guasta.

V 9.4.2 Esempio: configurazione della replica interregionale nativa HA in Kubernetes

Questo esempio implementa un gestore di code utilizzando la funzione nativa di replica interregionale ad alta disponibilità (Native HA CRR) nell' Kubernetes, utilizzando direttamente le risorse YAML.

Prima di iniziare

Per completare questo esempio, devi prima aver completato i seguenti prerequisiti:

- Creare uno spazio dei nomi (Kubernetes) in ciascuno dei cluster in cui verranno distribuiti i gruppi Live e Recovery.
- Sulla riga di comando, essere in grado di accedere a uno dei due cluster Kubernetes e passare allo spazio dei nomi pertinente.
- Assicurati di aver seguito i passaggi in “Preparare l'uso di 'Kubernetes creando un segreto di pull” a pagina 86 negli spazi dei nomi sopra indicati.

Informazioni su questa attività

Questo esempio utilizza due queue manager con la funzione Native HA CRR in un Kubernetes, utilizzando direttamente le risorse YAML. Uno dei gestori di coda Native HA sarà Live e l'altro sarà Recovery. Questo esempio si basa sui file YAML creati in “Esempio: Configurazione dell'HA nativo in 'Kubernetes” a pagina 176.

Questo esempio fornisce le modifiche richieste oltre a quelle definite per NativeHA: ConfigMap, Service, StatefulSet, e Route.

Procedura

1. Creare una nuova mappa di configurazione o definire una nuova definizione di dati.ini all'interno dell' ConfigMap e esistente (vedere il passaggio 2 nell'esempio Native HA), per definire il file INI per un gestore di code Live o Recovery.

   Creare un file Kubernetes ConfigMap contenente le definizioni per configurare i seguenti gruppi:

   a) Un gruppo nativo ad alta disponibilità come Recovery:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: example-nativeha-configmap
data:
...
  example-crr: |
    NativeHALocalInstance:
      GroupName=beta
```

```
      GroupRole=Recovery
      GroupLocalAddress=(9415)
      GroupCipherSpec=ANY_TLS12_OR_HIGHER
```

Quando viene distribuito il gruppo Recovery, è possibile determinare l'indirizzo di replica di tale gruppo e fornirlo nel ConfigMap per il gruppo Live. Ad esempio, questo può essere fatto impartendo il seguente comando al cluster di ripristino distribuito:

```
kubectl get route example-ibm-mq-crr-route -n <recovery-qm-namespace> -o
jsonpath="{.spec.host}"
```

b) Un gruppo di Native HA CRR come Live (indicando il gruppo Recovery):

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: example-nativeha-configmap
data:
...
  example-crr: |
    NativeHALocalInstance:
      GroupName=alpha
      GroupRole=Live
      GroupLocalAddress=(9415)
      GroupCipherSpec=ANY_TLS12_OR_HIGHER
    NativeHARecoveryGroup:
      Enabled=Yes
      GroupName=beta
      ReplicationAddress=<address.to.recovery.cluster>(443)
```

2. Esporre una porta di servizio aggiuntiva (ClusterIP) per il traffico Native HA CRR da far confluire nel gruppo CRR. Ciò può essere fatto creando una nuova definizione YAML o estendendo la definizione del servizio nel passaggio 3 dell'esempio Native HA:

```
apiVersion: v1
kind: Service
metadata:
...
  name: exampleqm-ibm-mq
spec:
  ports:
...
  - name: ha-crr
    port: 9415
    protocol: TCP
    targetPort: 9415
...
  selector:
...
  type: ClusterIP
```

3. Definire un nuovo percorso nel cluster per consentire la comunicazione dall'altro gruppo Native HA CRR per raggiungere questo gruppo Native HA CRR e abilitare il "pass-through" TLS alla nuova porta Service ClusterIP definita nel passaggio 2:

```
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  name: "exampleqm-ibm-mq-crr-route"
spec:
  to:
    kind: Service
    name: exampleqm-ibm-mq
  port:
    targetPort: 9415
  tls:
    termination: passthrough
```

4. Modificare l' StatefulSet e creato nel passaggio 6 dell'esempio Native HA, per garantire che siano soddisfatti i seguenti requisiti:

a) La nuova definizione INI di Native HA CRR dell' ConfigMap e è montata in modo che il gestore della coda possa rilevarla all'avvio:

```
...
        volumeMounts:
...
        # Mount the NativeHA CRR INI file to be applied when the queue manager starts
        - mountPath: /etc/mqm/example-crr.ini
          name: cm-example-nativeha-configmap
          readOnly: true
          subPath: example-crr.ini
...
      volumes:
...
      - configMap:
          defaultMode: 420
          items:
...
          - key: example-crr.ini
            path: example-crr.ini
          name: example-nativeha-configmap
        name: cm-example-nativeha-configmap
```

b) La comunicazione tra i due gruppi Native HA CRR deve essere criptata. Pertanto, le dichiarazioni della chiave segreta TLS per l'attuale gruppo CRR di NativeHA, e il certificato di fiducia per l'altro gruppo CRR di NativeHA, devono essere definite e montate nella directory `/etc/mqm/groupha` .

Utilizzando i nomi dei gruppi alfa (per Live) e beta (per Recovery), le modifiche di esempio per l' StatefulSet e YAML per il gruppo Live sarebbero:

```
...
        volumeMounts:
...
          - mountPath: /etc/mqm/groupha/pki/keys/ha-group
            name: groupha-tls
            readOnly: true
          - name: groupha-tls-trust-0
            mountPath: /etc/mqm/groupha/pki/trust/0
            readOnly: true
...
      volumes:
...
      - name: groupha-tls
        secret:
          secretName: nha-crr-secret-alpha
          defaultMode: 288
      - name: groupha-tls-trust-0
        secret:
          secretName: nha-crr-secret-beta
          defaultMode: 420
          items:
            - key: tls.crt
              path: native-crr-nha-crr-secret-beta-0-tls.crt
```

5. Verificare che i gruppi Native HA CRR siano in esecuzione, sincronizzati, connessi e in quorum, utilizzando il seguente comando **dspmq** sui pod Ready and Running dell'istanza del gestore delle code Live and Recovery:

```
kubectl exec -t <qmgr_pod_name> -- dspmq -o nativeha -g -m EXAMPLEQM
```

Lo stato dovrebbe mostrare che i due gruppi Native HA CRR sono collegati e agiscono come gruppi Live e Recovery, ad esempio:

```
QMNAME(EXAMPLEQM) ROLE(Active) INSTANCE(testmqqm-ibm-mq-0) INSYNC(yes) QUORUM(3/3)
GRPLSN(<0:0:10:30665>) GRPNAME(alpha) GRPROLE(Live)
 GRPNAME(alpha) GRPROLE(Live) GRPADDR(Unknown) GRPVER(9.4.2.0) GRSTATUS(Normal)
RCOVLSN(<0:0:10:30665>) RCOVTIME(2025-02-17T16:25:16.527089Z) INITLSN(<0:0:9:4684>)
INITTIME(2025-02-17T16:25:01.226650Z) LIVETIME(2025-02-17T16:25:02.974677Z)
ALTDATE(2025-02-17) ALTTIME(16.25.19)
 GRPNAME(beta) GRPROLE(Recovery) GRPADDR(<address.to.recovery.cluster>) GRPVER(9.4.2.0)
CONNGRP(yes) GRSTATUS(Normal) RCOVLSN(<0:0:10:30665>) RCOVTIME(2025-02-17T16:25:16.527089Z)
BACKLOG(0) INSYNC(yes) SYNCTIME(2025-02-17T16:29:38.643916Z) ALTDATE(2025-02-17)
ALTTIME(16.29.29)
```

Risultati

Hai implementato con successo due gruppi di gestione code con elevata disponibilità nativa e autenticazione TLS all'interno e tra i gruppi. Hai anche verificato che i gruppi Live e Recovery siano collegati e che eseguano la replica tra regioni.

V 9.4.2 Kubernetes Configurazione di Native HA e Cross-Region Replication su gestori di code con licenza IBM MQ in Kubernetes

Da IBM MQ 9.4.2, IBM MQ i gestori di code con licenza possono essere configurati per utilizzare le funzionalità di Native HA Cross-Region Replication (CRR) aggiungendo una licenza supplementare a un gestore di code.

Prima di iniziare

Nota: Queste istruzioni si applicano solo se si stanno distribuendo i gestori di code su Kubernetes. Se è stato distribuito il sito IBM MQ Operator, vedere invece "Configurazione dell'HA nativo e della replica interregionale sui gestori di code con licenza IBM MQ utilizzando il software IBM MQ Operator" a pagina 115 .

- L'indirizzo IBM License Service è necessario per utilizzare le funzioni CRR nativo HA sui gestori di code con licenza IBM MQ . Per ulteriori informazioni, vedere Installazione di License Service sul cluster Kubernetes nella documentazione di IBM Cloud Pak foundational services .
- Sul sito `StatefulSet` che gestisce il pod Queue Manager, l'etichetta `metadata.labels.app.kubernetes.io/instance` deve essere impostata su un valore univoco.

Procedura

1. Alla riga di comando, accedere al cluster Kubernetes e passare al namespace in cui è distribuito il gestore di code.
2. Aggiungere le annotazioni pertinenti alle annotazioni del gestore di code.

   Si noti che le istruzioni di questo passo riguardano la modifica di un gestore di code esistente. Per i nuovi gestori di code, aggiungere le annotazioni alla creazione.

   Importante: L'aggiornamento delle annotazioni del gestore di code richiede il riavvio del pod.

   a) Identificare il sito `StatefulSet` del gestore di code che si desidera aggiornare:

   ```
   kubectl get statefulset
   ```

   b) Aprire il sito `StatefulSet` del gestore di code per modificarlo:

   ```
   kubectl edit statefulset <QUEUE-MANAGER-STATEFUL-SET>
   ```

   c) Aggiungere al sito `StatefulSet` l'annotazione relativa alla componente imponibile. Si noti che ci sono altre annotazioni sotto `spec.template.metadata`; modificate solo le seguenti annotazioni:

   - Per IBM MQ:

   ```
   spec:
     template:
       metadata:
         productID: c661609261d5471fb4ff8970a36bccea
         productName: IBM MQ
   ```

   - Per IBM MQ for Non-Production Environment:

   ```
   spec:
     template:
       metadata:
         productID: 151bec68564a4a47a14e6fa99266deff
         productName: IBM MQ for Non-Production Environment
   ```

d) Salva le tue modifiche. Sulla riga di comando viene visualizzato un messaggio simile al seguente:

```
statefulset.apps/<QUEUE-MANAGER-STATEFUL-SET> edited
```

e) Eliminare tutti i pod di queue manager gestiti dal sito `StatefulSet` che è stato modificato.

f) Ripetere questi passaggi per altri gestori di code, se necessario.

3. Aggiungere il CR `IBMLicensingDefinition` pertinente che aggiungerà annotazioni di licenza aggiuntive al gestore di code per le funzioni CRR Native HA.

a) Creare un file YAML `IBMLicensingDefinition` .

Negli esempi seguenti, modificare questi valori:

- <IBMLicensingDefinition-UNIQUE-NAME>: Un nome univoco per il CR `IBMLicensingDefinition` . Una buona pratica è quella di includere il nome del gestore delle code in questo nome.
- <QUEUE-MANAGER-NAMESPACE>: Lo spazio dei nomi in cui è distribuito il gestore di code.
- <QUEUE-MANAGER-NAME>: Il valore di `metadata.labels.app.kubernetes.io/instance` del gestore di code.
- Esempio di YAML per un CR `IBMLicensingDefinition` da usare in produzione:

```
apiVersion: operator.ibm.com/v1
kind: IBMLicensingDefinition
metadata:
  name: <IBMLicensingDefinition-UNIQUE-NAME>
  namespace: <QUEUE-MANAGER-NAMESPACE>
spec:
  action: cloneModify
  condition:
    metadata:
      labels:
        app.kubernetes.io/instance: <QUEUE-MANAGER-NAME>
  scope: cluster
  set:
    productID: 46a57a4a66ee4b904e9b5544b53a2ba2
    productName: IBM MQ Native HA and Cross-Region Replication Add-on
```

- Esempio di YAML per un CR `IBMLicensingDefinition` per uso non di produzione:

```
apiVersion: operator.ibm.com/v1
kind: IBMLicensingDefinition
metadata:
  name: <IBMLicensingDefinition-UNIQUE-NAME>
  namespace: <QUEUE-MANAGER-NAMESPACE>
spec:
  action: cloneModify
  condition:
    metadata:
      labels:
        app.kubernetes.io/instance: <QUEUE-MANAGER-NAME>
  scope: cluster
  set:
    productID: 525b1331a74d4a209d9f6e9c908e374e
    productName: IBM MQ Native HA and Cross-Region Replication Add-on for Non-
Production Env
```

b) Applicare il file YAML `IBMLicensingDefinition` :

```
kubectl apply -f <IBMLicensingDefinition-UNIQUE-NAME>.yaml
```

c) Ripetere questi passaggi per altri gestori di code, se necessario.

Kubernetes Distribuzione e configurazione dei gestori di code utilizzando il diagramma Helm di esempio

È possibile distribuire e configurare un gestore code su Kubernetes utilizzando il grafico Helm di esempio.

Informazioni su questa attività

Se non stai utilizzando Red Hat OpenShift Container Platform, il IBM MQ Operator non è supportato. Puoi utilizzare il grafico Helm di esempio per la distribuzione su altri tipi di cluster Kubernetes .

Procedura

- Per informazioni su come usare Helm per distribuire la propria immagine del contenitore 'IBM MQ, vedere Esempio di grafico Helm 'IBM MQ.

Riferimenti correlati

“Supporto per IBM MQ nei contenitori” a pagina 28
Non tutte le funzioni IBM MQ sono disponibili e supportate nello stesso modo nei contenitori.

Podman Distribuzione di un gestore di code su 'Podman

Questo esempio esegue un'immagine precostituita di 'IBM MQ Advanced Container, per distribuire un gestore di code in 'Podman. Lo stesso esempio può essere utilizzato anche con 'Docker.

Prima di iniziare

Per convalidare la configurazione TLS alla fine di questo esempio:

- Installa il client IBM MQ. Sono necessari i comandi '**amqsputc** e '**amqsgetc** per testare l'invio e la ricezione dei messaggi. Questi comandi possono essere installati come parte del client 'IBM MQ.
  - Per 'Windows e 'Linux, installare il client ridistribuibile 'IBM MQ per il proprio sistema operativo da 'https://ibm.biz/mq94redistclients.
  - Per il 'Mac, scaricare e impostare il 'IBM MQ MacOS Toolkit dal 'https://developer.ibm.com/tutorials/mq-macos-dev/.
- Si noti che non esiste un'immagine " ARM64 " IBM MQ Advanced Container precostituita.

Questo esempio utilizza il " Podman, ma può essere utilizzato anche con il " Docker.

Informazioni su questa attività

Questo esempio esegue un'immagine precostituita di 'IBM MQ Advanced Container, per distribuire un gestore di code in 'Podman. Vengono illustrati i passaggi aggiuntivi necessari per configurare TLS per la messaggistica e per configurare il 'IBM MQ Console e il 'REST API. È possibile scegliere quali configurare, saltando i passaggi non necessari per soddisfare le proprie esigenze.

Quando il gestore di code è in funzione, questo esempio utilizza i programmi di esempio di 'IBM MQ in esecuzione su una macchina esterna a 'Podman, come il vostro portatile, per convalidare l'uso di TLS reciproco tra il client di esempio e il gestore di code per mettere e ricevere messaggi. Inoltre, convalida l'accesso al 'IBM MQ Console e le chiamate amministrative e di messaggistica al 'REST API.

Nota: Si può invece eseguire un'immagine del contenitore " IBM MQ Advanced for Developers " precostruita, che ha alcune configurazioni predefinite aggiuntive per gli sviluppatori. Consultare “IBM MQ Advanced for Developers immagine contenitore” a pagina 291.

Procedura

- **Configurare TLS per la messaggistica tra un client e il gestore di code**
  a) Creare i certificati come descritto in '“Creazione di una PKI autofirmata utilizzando OpenSSL” a pagina 104, seguendo tutti i passaggi ma saltando il comando '**kubectl**.
    a. Creare una directory, ad esempio chiamata " `qm-pki`, contenente i file delle chiavi e dei certificati generati per il gestore delle code:
      i) `example-qm.key`
      ii) `example-qm.crt`

iii) `ca.crt`

L'utente che esegue il contenitore (uid 1001) ha bisogno di accedere in lettura a '`example-qm.key`. Per impostazione predefinita, 'OpenSSL la crea in modo che l'utente che crea la chiave abbia accesso ad essa.

b. Creare una cartella, ad esempio chiamata '`mq-client`, contenente i file generati per il client:

i) `example-app1.p12`

ii) Per il 'Mac è necessario anche il '`example-app1-chain.crt`

b) Creare un file contenente i comandi MQSC, ad esempio '**`example-tls.mqsc`**, per creare una nuova coda e un canale 'SVRCONN e per aggiungere un record di autenticazione del canale che consenta l'accesso al canale.

```
DEFINE CHANNEL('MTLS.SVRCONN') CHLTYPE(SVRCONN) SSLCAUTH(REQUIRED)
SSLCIPH('ANY_TLS13_OR_HIGHER') REPLACE
SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=*') USERSRC(NOACCESS)
ACTION(REPLACE)
SET CHLAUTH('MTLS.SVRCONN') TYPE(SSLPEERMAP) SSLPEER('CN=example-app1') USERSRC(MAP)
MCAUSER('app1') ACTION(REPLACE)
SET AUTHREC PRINCIPAL('app1') OBJTYPE(QMGR) AUTHADD(CONNECT,INQ)
DEFINE QLOCAL('EXAMPLE.QUEUE') REPLACE
SET AUTHREC PROFILE('EXAMPLE.QUEUE') PRINCIPAL('app1') OBJTYPE(QUEUE)
AUTHADD(BROWSE,PUT,GET,INQ)
```

Il comando MQSC definisce un canale chiamato 'MTLS.SVRCONN e una coda chiamata 'EXAMPLE.QUEUE. Il canale è configurato per consentire l'accesso solo ai client che presentano un certificato con un "nome comune" di '`example-app1`. Si tratta del nome comune utilizzato in uno dei certificati creati nel passaggio precedente. Le connessioni su questo canale con questo nome comune sono mappate su un ID utente di 'app1, che è autorizzato a connettersi al gestore di code e ad accedere alla coda di esempio.

c) Creare un file INI, ad esempio " `example-tls.ini`, che abiliti un criterio di sicurezza tale per cui l'ID utente " app1 non debba esistere in un registro utenti esterno: esiste solo come nome in questa configurazione.

```
Service:
    Name=AuthorizationService
    EntryPoints=14
    SecurityPolicy=UserExternal
```

- **Configure the IBM MQ Console and REST API**

  In questo esempio, si configura un registro di base all'interno del file '`mqwebuser.xml`. I nomi utente, le password e i ruoli contenuti nel file XML vengono utilizzati per autenticare e autorizzare gli utenti di 'IBM MQ Console e 'REST API. Per ulteriori informazioni, vedere Configurazione di un registro di base per 'IBM MQ Console e 'REST API.

  È necessario valutare se l'uso di un registro di base è appropriato per gli ambienti di produzione. Vedere ' sicurezza IBM MQ Console e 'REST API .

  a) Creare un file chiamato '`mqwebuser.xml`, per configurare il 'IBM MQ Console.

  In questo esempio viene creato un registro di base, contenente utenti con i ruoli di web admin e web user.

  È necessario sostituire *ADMIN_PASSWORD* e *APP_PASSWORD* con i propri valori. Se lo si desidera, è possibile modificare anche i nomi utente dell'amministratore e dell'applicazione, anche se il nome utente dell'applicazione deve corrispondere al nome utente dell'applicazione specificato nel comando MQSC precedente. Si noti che i nomi degli utenti sono utilizzati due volte nell'esempio.

```
<?xml version="1.0" encoding="UTF-8"?>
<server>
    <featureManager>
    <feature>appSecurity-2.0</feature>
    <feature>basicAuthenticationMQ-1.0</feature>
    </featureManager>
    <enterpriseApplication id="com.ibm.mq.console">
        <application-bnd>
```

```
            <security-role name="MQWebAdmin">
                <group name="MQWebAdminGroup" realm="defaultRealm"/>
            </security-role>
        </application-bnd>
    </enterpriseApplication>
    <enterpriseApplication id="com.ibm.mq.rest">
        <application-bnd>
            <security-role name="MQWebAdmin">
                <group name="MQWebAdminGroup" realm="defaultRealm"/>
            </security-role>
            <security-role name="MQWebUser">
                <group name="MQWebMessaging" realm="defaultRealm"/>
            </security-role>
        </application-bnd>
    </enterpriseApplication>
    <basicRegistry id="basic" realm="defaultRealm">
        <user name="admin" password="ADMIN_PASSWORD"/>
        <user name="app1" password="APP_PASSWORD"/>
        <group name="MQWebAdminGroup">
            <member name="admin"/>
        </group>
        <group name="MQWebMessaging">
            <member name="app1"/>
        </group>
    </basicRegistry>
    <sslDefault sslRef="mqDefaultSSLConfig"/>
</server>
```

b) Accedere al registro del contenitore.

Ottenere una chiave di abilitazione, quindi usarla per accedere al registro del contenitore. Consultare “Preparazione all'uso di 'Podman o 'Docker mediante l'estrazione dell'immagine del contenitore precostruito” a pagina 87.

c) Eseguire il gestore delle code in 'Podman.

Completare una delle seguenti opzioni:

– **Opzione 1: Eseguire il gestore delle code in 'Podman montando la directory contenente i certificati del gestore delle code.**

Il comando seguente include la configurazione della messaggistica TLS e la configurazione di 'IBM MQ Console e 'REST API. Se si è saltata una delle configurazioni descritte in precedenza, perché non necessaria per le proprie esigenze, rimuovere i parametri relativi alle sezioni saltate.

Assicurarsi che il comando punti alla posizione corretta dei file e della directory creati in precedenza.

Per eseguire il gestore di code in 'Podman, eseguire il seguente comando:

```
podman run --env MQ_ENABLE_EMBEDDED_WEB_SERVER=true  \
-v "./mqwebuser.xml:/etc/mqm/web/installations/Installation1/servers/mqweb/
mqwebuser.xml" \
-v "./example-tls.ini:/etc/mqm/example-tls.ini"  \
-v "./example-tls.mqsc:/etc/mqm/example-tls.mqsc"  \
-v "./qm-pki:/etc/mqm/pki/keys/qmlabel" --env LICENSE=accept --env MQ_QMGR_NAME=QM1 \
--publish 1414:1414 --publish 9443:9443 --detach --name QM1 cp.icr.io/cp/ibm-
mqadvanced-server:9.4.1.0-r1
```

Il comando estrae l'immagine del gestore di code 'IBM MQ Advanced e avvia il contenitore.

– **Opzione 2: Eseguire il gestore di code in 'Podman utilizzando i segreti di 'Podman**

È possibile montare i certificati del gestore di code usando i segreti 'Podman, invece di montare direttamente le directory che li contengono nel contenitore.

Nota: Questa opzione non si applica quando si usa il 'Docker, a meno che non si usi lo Sciame 'Docker, perché i segreti 'Docker possono essere usati solo con lo Sciame 'Docker. Vedere il comando docker secret create in dockerdocs.

Completare i seguenti passi:

a. Creare i segreti 'Podman per ogni file TLS del gestore delle code. Ad esempio, dalla cartella di creazione 'qm-pki:

```
podman secret create example-qm.key ./example-qm.key
podman secret create example-qm.crt ./example-qm.crt
podman secret create ca.crt ./ca.crt
```

Il comando seguente include la configurazione della messaggistica TLS e la configurazione di 'IBM MQ Console e 'REST API. Se si è saltata una delle configurazioni descritte in precedenza, perché non necessaria per le proprie esigenze, rimuovere i parametri relativi alle sezioni saltate.

Assicurarsi che il comando punti alla posizione corretta dei file e della directory creati in precedenza.

Per eseguire il gestore di code in 'Podman, eseguire il seguente comando:

```
podman run --env MQ_ENABLE_EMBEDDED_WEB_SERVER=true \
-v "./mqwebuser.xml:/etc/mqm/web/installations/Installation1/servers/mqweb/
mqwebuser.xml" \
-v "./example-tls.ini:/etc/mqm/example-tls.ini" \
-v "./example-tls.mqsc:/etc/mqm/example-tls.mqsc" \
--secret example-qm.key,target=/etc/mqm/pki/keys/qmlabel/example-
qm.key,uid=1001,mode=0440 \
--secret example-qm.crt,target=/etc/mqm/pki/keys/qmlabel/example-qm.crt,uid=1001 \
--secret ca.crt,target=/etc/mqm/pki/keys/qmlabel/ca.crt,uid=1001 --env
LICENSE=accept --env MQ_QMGR_NAME=QM1 \
--publish 1414:1414 --publish 9443:9443 --detach --name QM1 cp.icr.io/cp/ibm-
mqadvanced-server:9.4.1.0-r1
```

Nota: Se l'utente con cui viene eseguito il contenitore è stato sovrascritto a un utente diverso, modificare l'UID in modo che corrisponda.

Un gestore di code è in esecuzione in 'Podman.

```
podman ps
```

Convalida

- **Convalidare la messaggistica con TLS**

a) Creare un file INI del client, chiamato 'mqclient.ini, nella directory del client creata in precedenza, ad esempio 'mq-client. Specificare la *PASSWORD* utilizzata per il keystore p12 al punto 1 di Configurazione di TLS per la messaggistica tra un client e il gestore di code:

```
Channels:
  ChannelDefinitionDirectory=.
  ChannelDefinitionFile=CCDT.json
SSL:
  OutboundSNI=HOSTNAME
  SSLKeyRepository=example-app1.p12
  SSLKeyRepositoryPassword=PASSWORD
```

b) Nella stessa directory, creare un file CCDT con un nome che corrisponda al riferimento del file 'mqclient.ini precedentemente definito, ad esempio 'CCDT.json:

```
{
    "channel":
    [
        {
            "name": "MTLS.SVRCONN",
            "clientConnection":
            {
                "connection":
                [
                    {
                        "host": "localhost",
                        "port": 1414
                    }
                ],
                "queueManager": "QM1"
            },
            "transmissionSecurity":
```

```
            {
              "cipherSpecification": "ANY_TLS13",
              "certificateLabel": "example-app1"
            },
            "type": "clientConnection"
        }
    ]
}
```

c) Su 'Mac, eseguire il seguente comando:

```
export MQSSLTRUSTSTORE=example-app1-chain.crt
```

d) All'interno della stessa directory, ad esempio '`mq-client`, utilizzate l'applicazione di esempio per inserire un messaggio:

```
amqsputc EXAMPLE.QUEUE QM1
```

e) Utilizzare l'applicazione di esempio per ottenere un messaggio:

```
amqsgetc EXAMPLE.QUEUE QM1
```

- **Convalidare il 'IBM MQ Console**

  Collegarsi alla console in un browser:

  a) Apertura: https://localhost:9443/ibmmq/console

  b) Accettare il certificato autofirmato.

  c) Effettuare il login:

  – Utente: `admin`
  – Password: il valore precedentemente specificato per *ADMIN_PASSWORD*.

- **Convalidare il 'REST API**

  Negli esempi seguenti, specificare i valori precedentemente configurati per *ADMIN_PASSWORD* e *APP_PASSWORD*.

  Un esempio di chiamata al 'administrative REST API per ottenere lo stato del gestore di code:

```
curl -k -X GET -u admin:ADMIN_PASSWORD https://localhost:9443/ibmmq/rest/v1/admin/qmgr/QM1
```

  Un esempio di chiamata della messaggistica 'REST API per inserire e poi prelevare un messaggio da una coda.

  Inserite un messaggio:

```
curl -k -X POST -u app1:APP_PASSWORD \
-H 'Content-Type: application/json' -H "ibm-mq-rest-csrf-token: value" \
-d 'msg from REST' https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/
EXAMPLE.QUEUE/message
```

  Sfogliare un messaggio:

```
curl -k -X GET -u app1:APP_PASSWORD \
-H "ibm-mq-rest-csrf-token: value" https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/
queue/EXAMPLE.QUEUE/message
```

  Nota: L'operazione 'REST API '**GET** è in realtà un'operazione di browse 'IBM MQ: il messaggio viene lasciato in coda. Per fare un 'IBM MQ '**GET**, usare il verbo 'REST API '**DELETE**.

Risoluzione dei problemi

- Se le applicazioni di esempio '**amqputc** o '**amqgetc** falliscono, visualizzare i log degli errori in '`~/IBM/MQ/data/errors/AMQERR01.LOG` sul computer client, ad esempio il portatile.
- Visualizzare i log degli errori del gestore delle code nei loro luoghi abituali, eseguendo il container in 'Podman.

  Ad esempio:

```
podman exec -it QM1 /bin/bash
```

OpenShift CD CP4I-SC2 Migrazione a IBM MQ Operator

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 Red Hat OpenShift Container Platform.

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 194
2. “Estrazione della configurazione del gestore code” a pagina 194
3. Opzionale: “Facoltativo: estrazione e acquisizione delle chiavi e dei certificati del gestore code” a pagina 195
4. Opzionale: “Facoltativo: configurazione di LDAP” a pagina 197
5. Opzionale: “Facoltativo: modifica degli indirizzi IP e dei nomi host nella configurazione IBM MQ” a pagina 205
6. “Aggiornamento della configurazione del gestore code per un ambiente contenitore” a pagina 206
7. “Selezione dell'architettura HA di destinazione per IBM MQ in esecuzione nei contenitori” a pagina 209
8. “Creazione delle risorse per il gestore code” a pagina 209
9. “Creazione del nuovo gestore code su Red Hat OpenShift” a pagina 210

10. “Verifica della nuova distribuzione del contenitore” a pagina 214

OpenShift CD CP4I-SC2 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 a IBM MQ Operator” a pagina 193.

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 28.
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 159.
4. Se il sistema IBM MQ include l'alta disponibilità, esaminare le opzioni disponibili.

   Consultare “Pianificazione dell'alta disponibilità per IBM MQ nei contenitori” a pagina 40.

Operazioni successive

È ora possibile estrarre la configurazione del gestore code.

OpenShift CD CP4I-SC2 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 CD CP4I-SC2 Facoltativo: estrazione e acquisizione delle chiavi e dei certificati del gestore code

IBM MQ può essere configurato per codificare il traffico di rete nel gestore code con TLS. Utilizzare questa attività per verificare che il gestore code stia 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 certi che la comunicazione TLS sia configurata 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 .

Nota: I certificati del gestore code con lo stesso DN (Distinguished Name) dell'emittente (CA) non sono supportati. Un certificato deve avere un DN (Distinguished Name) soggetto univoco. Il prodotto verifica che i DN non siano uguali.

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 runmqakmResult < <(runmqakm -cert -list -db ${keyrlocation}.kdb -stashed)
   cert=1
   for i in "${runmqakmResult[@]:2}"
   do
      certlabel=$(echo ${i:2} | xargs)
      echo Extracting certificate $certlabel to $cert.cert
      runmqakm -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 del repository delle chiavi TLS del gestore code:

  ```
  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:

  ```
  runmqakm -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:

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

  Negli esempi precedentemente visualizzati, ciò equivale ai seguenti comandi:

  ```
  runmqakm -cert -extract -db /run/runmqserver/tls/key.kdb -label "CN=cs-ca-
  certificate,O=cert-manager" -target /tmp/cert-manager.crt -stashed
  runmqakm -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 CD CP4I-SC2 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.

OpenShift CD CP4I-SC2 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 197.

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:

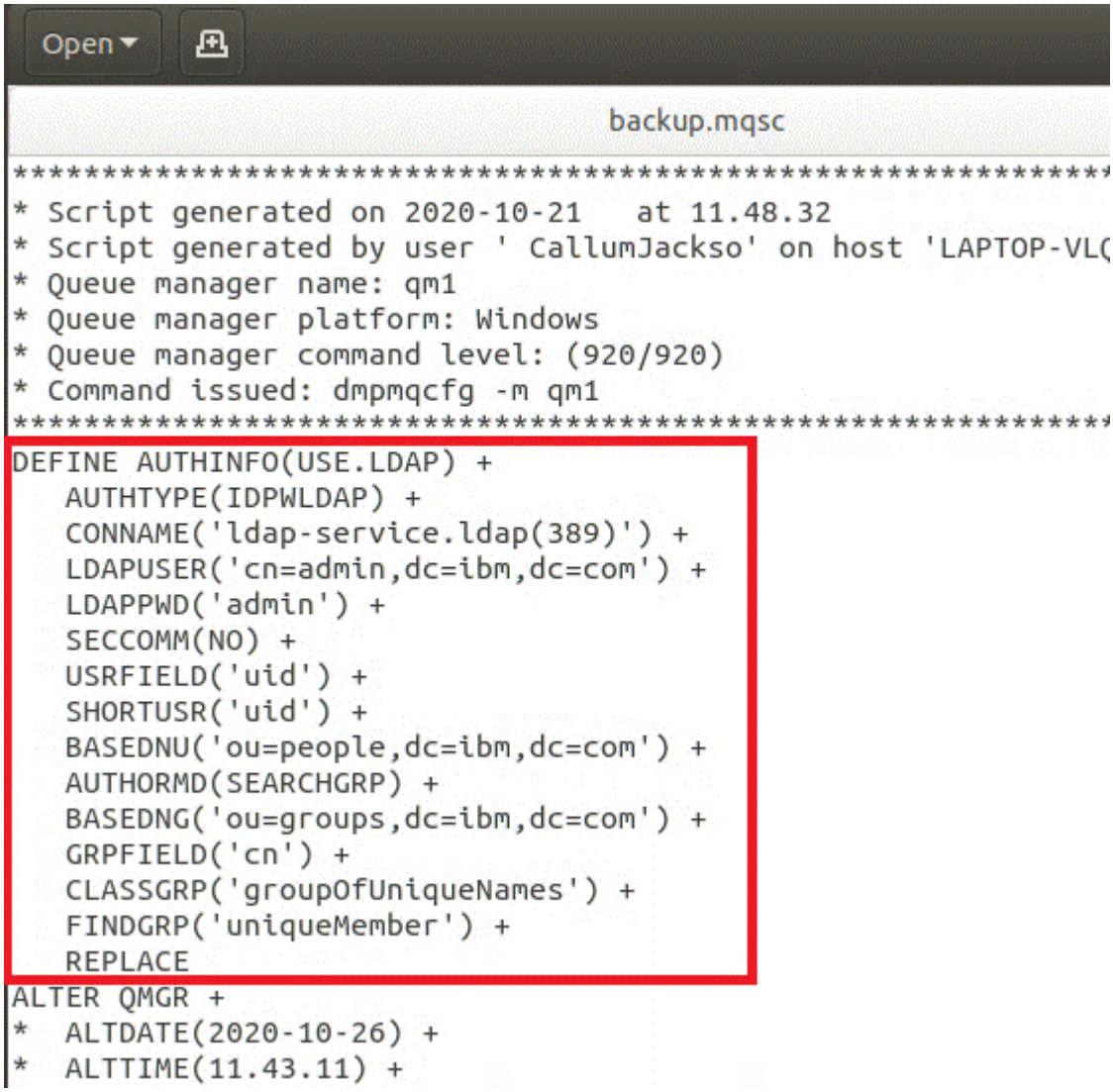
```
DEFINE AUTHINFO(USE.LDAP) +
   AUTHTYPE(IDPWLDAP) +
   CONNAME('ldap-service.ldap(389)') +
   LDAPUSER('cn=admin,dc=ibm,dc=com') +
   LDAPPWD('admin') +
   SECCOMM(NO) +
   USRFIELD('uid') +
   SHORTUSR('uid') +
   BASEDNU('ou=people,dc=ibm,dc=com') +
   AUTHORMD(SEARCHGRP) +
   BASEDNG('ou=groups,dc=ibm,dc=com') +
   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 CD CP4I-SC2 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 197.

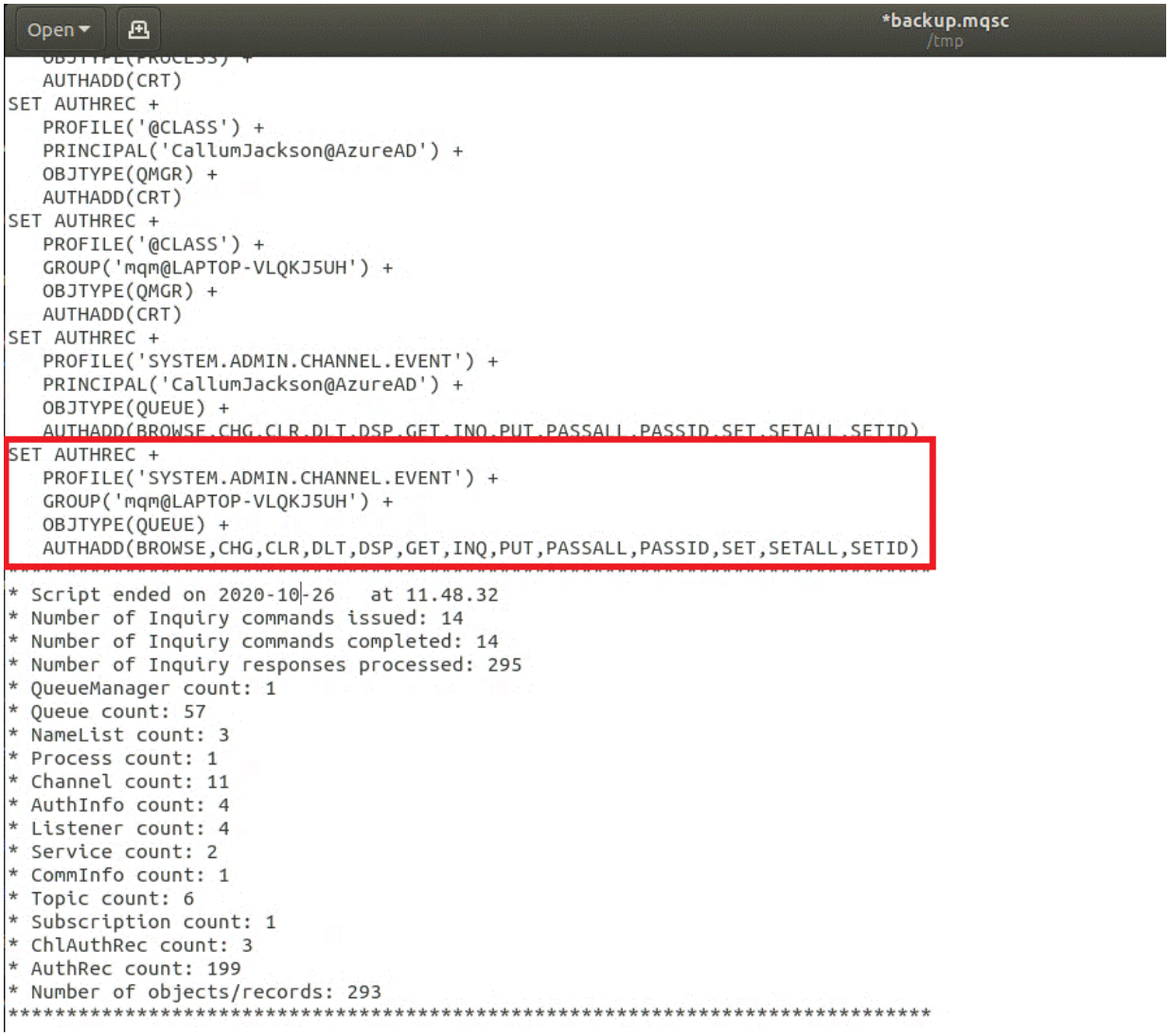
Esistono due parti per aggiornare le informazioni di autorizzazione LDAP:

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 CD CP4I-SC2 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.

OpenShift CD CP4I-SC2 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.

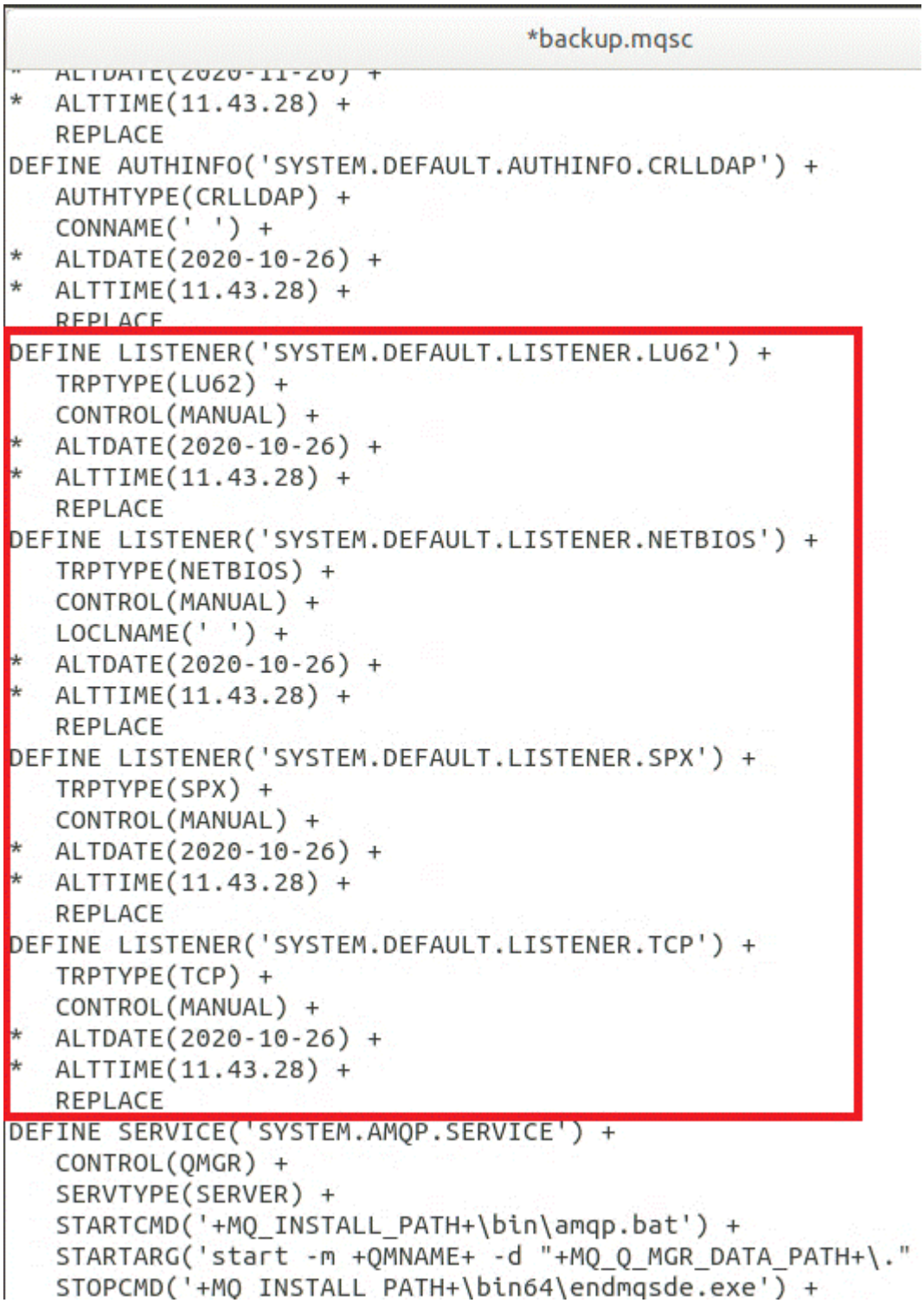
Pertanto, è necessario aggiornare la configurazione esportata:

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.



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 CD CP4I-SC2 Selezione dell'architettura HA di destinazione per IBM MQ in esecuzione nei contenitori

Scegli tra una singola istanza (un singolo pod Kubernetes), più istanze (due pod) e la HA nativa (un pod di replica attivo e due pod di replica standby) 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 tre 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 .
- **HA nativa**: tre contenitori (ciascuno in un pod separato), ciascuno con una 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. Se il pod che esegue 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.

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 210).

Procedura

1. Esamina le tre opzioni.

   Per una descrizione completa di queste opzioni, vedere “Pianificazione dell'alta disponibilità per IBM MQ nei contenitori” a pagina 40.
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.

OpenShift CD CP4I-SC2 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 195, 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 CD CP4I-SC2 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 209, ci sono tre possibili topologie di distribuzione. Pertanto, questo argomento fornisce tre diversi modelli:

- Modello 1: distribuire un gestore code a istanza singola.
- Modello 2: distribuire un gestore code a più istanze.
- Modello 3: distribuire un gestore code HA nativo.

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

Procedura

- **Modello 1: distribuire un singolo gestore code dell'istanza.**

  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:

  Nota: Questo esempio utilizza una licenza IBM MQ Advanced , ma è possibile utilizzare anche una licenza IBM MQ . Per ulteriori informazioni, consultare “Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator” a pagina 160.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: qm1
spec:
  version: 9.4.3.0-r1
  license:
    accept: true
    license: L-CYPF-CRPF3H
    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: 9.4.3.0-r1
```

```
  license:
    accept: true
    license: L-CYPF-CRPF3H
    use: "Production"
```

Ciò corrisponde alle informazioni sulla versione e sulla licenza per la distribuzione. Se è necessario personalizzarlo, utilizzare le informazioni fornite in “mq.ibm.com/v1beta1: Versioni di licenza attuali” a pagina 248.

```
  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.

- **Modello 2: distribuzione di 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.

  Nota: Questo esempio utilizza una licenza IBM MQ Advanced , ma è possibile utilizzare anche una licenza IBM MQ . Per ulteriori informazioni, consultare “Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator” a pagina 160.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: qm1mi
spec:
  version: 9.4.3.0-r1
  license:
    accept: true
    license: L-CYPF-CRPF3H
    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.

- **Modello 3: distribuire un gestore code HA nativo.**

  Per un esempio di creazione di un gestore code HA nativo, consultare “Esempio: configurazione della HA nativa utilizzando IBM MQ Operator” a pagina 112.

OpenShift CD CP4I-SC2 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 210. 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 a IBM MQ Operator” a pagina 193.

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 148.

Funzionamento di IBM MQ nei contenitori

Se hai bisogno di operare o interagire con i gestori code IBM MQ in esecuzione nei contenitori, consulta i seguenti argomenti per ulteriori informazioni.

Procedura

- “Utilizzo di IBM MQ mediante IBM MQ Operator” a pagina 216.
- “Visualizzazione dello stato dei gestori code della HA nativa” a pagina 229.
- “Conclusione dei gestori di coda HA nativi” a pagina 239.

OpenShift CP4I Utilizzo di IBM MQ mediante IBM MQ Operator

Procedura

- “Connessione al IBM MQ Console distribuito in un cluster Red Hat OpenShift” a pagina 217.
- “Monitoraggio quando si utilizza IBM MQ Operator” a pagina 217.
- “Backup e ripristino della configurazione del gestore code utilizzando la CLI Red Hat OpenShift” a pagina 229.

OpenShift CP4I Concessione di autorizzazioni per IBM MQ Console

Le autorizzazioni per IBM MQ Console sono gestite in modo diverso in base al tuo uso della licenza.

Informazioni su questa attività

Scegliere l'opzione pertinente in base alla licenza utilizzata per il gestore di code.

Procedura

- **Opzione 1:** Configurare utenti e ruoli tramite Keycloak.

  Se stai utilizzando una licenza IBM Cloud Pak for Integration , IBM MQ Console utilizza Keycloak per la gestione dell'identità e dell'accesso.

  – Per informazioni sulla configurazione Keycloak, vedere Gestione dell'identità e dell'accesso nella documentazione IBM Cloud Pak for Integration.
  – Per informazioni sui *ruoliMQ* e sui ruoli Keycloak corrispondenti, vedere i ruoli e le autorizzazioni diCloud Pak for Integration nella documentazione IBM Cloud Pak for Integration I ruoli possono essere configurati per l'intera installazione o per ogni singola istanza (queue manager). Gli equivalenti *dei ruoliMQ* sono elencati in **Queue manager**.
  – Se in precedenza si sono configurati gli utenti con IAM su versioni precedenti di IBM MQ Operator, vedere Migrazione degli utenti da IAM a Keycloak nella documentazione di IBM Cloud Pak for Integration.
- **Opzione 2:** Configurare manualmente gli utenti e i ruoli.

Se si utilizza una licenza IBM MQ, la IBM MQ Console non è preconfigurata ed è necessario configurarla da soli.

- Per informazioni su utenti e ruoli, vedere Configurazione di utenti e ruoli.
- Per un semplice esempio, vedere “Configurazione di IBM MQ Console con un registro di base utilizzando IBM MQ Operator” a pagina 165.
- In alternativa, se si modifica un gestore di code per utilizzare una licenza IBM Cloud Pak for Integration, è possibile configurare Keycloak come descritto in precedenza.

OpenShift CP4I Connessione al IBM MQ Console distribuito in un cluster Red Hat OpenShift

Come connettersi al IBM MQ Console di un gestore code distribuito su un cluster Red Hat OpenShift Container Platform .

Prima di iniziare

È necessario avere l'autorizzazione per accedere al sito IBM MQ Console. Vedere “Concessione di autorizzazioni per IBM MQ Console” a pagina 216.

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 UI. In alternativa, è possibile trovarla dalla Red Hat OpenShift CLI eseguendo il seguente comando:

```
oc get queuemanager QueueManager Name -n namespace_of_your_MQ_deployment --output
jsonpath='{.status.adminUiUrl}'
```

Attività correlate

“Configurazione di un instradamento per la connessione a un gestore code dall'esterno di un cluster Red Hat OpenShift” a pagina 148
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 Monitoraggio quando si utilizza IBM MQ Operator

I gestori code gestiti da IBM MQ Operator possono produrre metriche compatibili con Prometheus. Le metriche del gestore code sono abilitate e servite tramite HTTP per impostazione predefinita.

Prima di iniziare

Utilizzo di HTTPS per le metriche del gestore code

Da 9.4.2.0-r1, i responsabili della coda possono fornire metriche tramite HTTPS utilizzando un certificato creato dall' OpenShift . Prima dell' 9.4.2.0-r1, i gestori delle code possono fornire solo metriche tramite HTTP.

Nota: Tutti i gestori di code forniscono metriche tramite l' HTTP, per impostazione predefinita.

Informazioni su questa attività

Prometheus è un database open source di serie temporali e un motore di valutazione delle regole per le metriche. I contenitori dell' IBM MQ e espongono un endpoint metrico che può essere interrogato da Prometheus. Le metriche sono generate dagli argomenti del sistema di gestione delle attività (IBM MQ Operator) per il monitoraggio e la tracciatura delle attività.

OpenShift Container Platform include uno stack di monitoraggio preconfigurato, preinstallato e autoaggiornante che utilizza un server di gestione dei dispositivi (Prometheus). Lo stack di controllo OpenShift Container Platform deve essere configurato per monitorare i progetti definiti dall'utente. Vedere Abilitazione del monitoraggio per progetti definiti dall'utente.

Quando si crea un `QueueManager` con le metriche abilitate, l' IBM MQ Operator e crea un `ServiceMonitor` che l' Prometheus e può poi scoprire.

Procedura

- Visualizza le metriche del gestore code utilizzando lo stack di monitoraggio Red Hat OpenShift Container Platform (OCP).
  a) Aprire **la scheda Metriche** in OCP.
  b) Fare clic **su Osserva** > **metriche**.
- Servire le metriche del gestore della coda attraverso HTTPS

  Impostare **`.spec.queueManager.metrics.tls.provider`** su `openshift`.

  ```
  spec:
    queueManager:
      metrics:
        enabled: true
        tls:
          provider: openshift
  ```

  Nota:

  Utilizzando questa impostazione si crea un certificato di autenticazione (OpenShift) che viene utilizzato dal server di metriche (HTTPS) di un gestore di code. Nessun altro fornitore può essere configurato per l'uso da parte dell' IBM MQ Operator.
- Disabilitare le metriche del gestore code.

  Impostare **`.spec.queueManager.metrics.enabled`** su `false`.

Attività correlate

“Troubleshooting container clusters with the OpenShift Monitoring CLI” a pagina 245
As a troubleshooting aid, you can query the available container cluster metrics using the OpenShift Monitoring CLI.

OpenShift CP4I *Monitoring container clusters with the OpenShift Monitoring console*

You can query the available container cluster metrics using the OpenShift Monitoring web console.

Before you begin

OpenShift Monitoring provides a web console and a CLI. For day-to-day operations you might prefer to use the console, as described in this task. For gathering troubleshooting information you might prefer to use the CLI, as described in “Troubleshooting container clusters with the OpenShift Monitoring CLI” on page 245.

About this task

OpenShift Container Platform includes a pre-configured, pre-installed and self-updating monitoring stack that uses an open source Prometheus server. Queue managers running under an IBM MQ Operator can produce metrics compatible with Prometheus. For a complete list of the metrics available for IBM MQ, see “Metriche pubblicate dall' IBM MQ ” on page 220.

OpenShift Monitoring is the default cluster monitoring method. It is installed with the cluster. It provides Prometheus, Thanos Querier, and a Grafana dashboard that shows queue manager metrics.

To query the available container cluster metrics using the OpenShift Monitoring console, complete the following steps.

Procedure

1. Enable OpenShift Monitoring for the operands deployed in an IBM MQ container cluster.
   a) Log in to the cluster as a cluster administrator.
   b) Add "`enableUserWorkload: true`" to the configMap `cluster-monitoring-config` in the `openshift-monitoring` namespace. To do this, follow the steps in Enabling monitoring for user-defined projects in the Red Hat OpenShift documentation. If the `cluster-monitoring-config` ConfigMap does not exist in the `openshift-monitoring` namespace, follow the instructions to create it.
2. Query the IBM MQ metrics using the OpenShift Monitoring console.

   Use the static Grafana dashboard, which presents cluster metrics to cluster administrators.

   Query IBM MQ metrics, and examine the data plotted on a chart, as described in Querying metrics in the Red Hat OpenShift documentation.

   For example, if you query the metric **`ibmmq_qmgr_cpu_load_fifteen_minute_average_percentage`** you get a chart similar to the following screen capture:

   This screen capture also shows the web console prompt. You can run PromQL queries from this prompt. For example:

   Get a list of queue managers that have a positive FDC count:

```
ibmmq_qmgr_fdc_files > 0
```

   Pass the `namespace` filter in the query to get the results for a specific namespace:

```
ibmmq_qmgr_fdc_files{namespace='your-namespace'} > 0
```

   Return the count of all such queue managers:

```
count(ibmmq_qmgr_fdc_files{namespace='your-namespace'} > 0)
```

   Get the Write latency of queue manager pods over the last 5 hours:

```
ibmmq_qmgr_log_write_latency_seconds{namespace='your-namespace'}[5h]
```

Get the RAM usage estimate in bytes for a queue manager for a particular namespace summed by "Job":

```
sum by (job)(ibmmq_qmgr_ram_usage_estimate_for_queue_manager_bytes{qmgr='your-queue-manager', namespace='your-namespace'})
```

The previous queries illustrate use of filters "namespace", "job" and "qmgr". For this metric you can also filter on "container", "endpoint", "instance", "pod", and "prometheus". Note that the available filters are the column headings shown in the screen capture, except for “Name” which is the name of the metric itself.

Related tasks

“Troubleshooting container clusters with the OpenShift Monitoring CLI” on page 245
As a troubleshooting aid, you can query the available container cluster metrics using the OpenShift Monitoring CLI.

Thanos Querier Versus Thanos Querier

Related reference

“Metriche pubblicate dall' IBM MQ ” on page 220
I contenitori dei gestori code possono pubblicare metriche compatibili con Red Hat OpenShift Monitoring.

Metriche pubblicate dall' IBM MQ

I contenitori dei gestori code possono pubblicare metriche compatibili con Red Hat OpenShift Monitoring.

Metrica	Tipo	Descrizione
ibmmq_nha_recovery_average_network_round_trip_time_seconds	gauge	Tempo medio di andata e ritorno della rete
ibmmq_nha_recovery_backlog_average_bytes	gauge	Byte medi di backlog
ibmmq_nha_recovery_backlog_bytes	gauge	Byte di backlog
ibmmq_nha_recovery_compressed_log_sent_bytes	gauge	Byte di log compressi inviati
ibmmq_nha_recovery_log_data_average_compression_time_seconds	gauge	Tempo medio di compressione dei dati di log
ibmmq_nha_recovery_log_data_average_decompression_time_seconds	gauge	Tempo medio di decompressione dei dati di log
ibmmq_nha_recovery_log_decompressed_bytes	gauge	Byte di log decompressi
ibmmq_nha_recovery_log_sent_bytes	counter	Byte di log inviati
ibmmq_nha_recovery_rebase_count	gauge	Conteggio dei rebase

Metrica	Tipo	Descrizione
ibmmq_nha_recovery_recovery_log_sequence_number	gauge	Numero di sequenza del log di ripristino
ibmmq_nha_replication_acknowledged_log_sequence_number_total	gauge	Numero di sequenza di log riconosciuto
ibmmq_nha_replication_average_network_round_trip_time_seconds	gauge	Tempo medio di andata e ritorno della rete
ibmmq_nha_replication_backlog_average_bytes	gauge	Byte medi di backlog
ibmmq_nha_replication_backlog_bytes	gauge	Byte di backlog
ibmmq_nha_replication_catch_up_compressed_log_sent_bytes	counter	Byte di log compressi di catch-up inviati
ibmmq_nha_replication_catch_up_log_data_average_compression_time_seconds	gauge	Tempo medio di compressione dei dati di log di catch-up
ibmmq_nha_replication_catch_up_log_data_average_decompression_time_seconds	gauge	Tempo medio di decompressione dei dati di log di catch-up
ibmmq_nha_replication_catch_up_log_decompressed_bytes	gauge	Byte di log di catch-up decompressi
ibmmq_nha_replication_catch_up_uncompressed_log_sent_bytes	counter	Byte di log non compressi di catch-up inviati
ibmmq_nha_replication_catchup_log_sent_bytes	counter	Byte log Catch-up inviati
ibmmq_nha_replication_log_file_system_free_space_percent	gauge	File system di log - spazio disponibile
ibmmq_nha_replication_log_file_system_in_use_bytes	gauge	File system log - byte in uso

Metrica	Tipo	Descrizione
ibmmq_nha_replication_log_write_average_acknowledgement_latency_seconds	gauge	Latenza riconoscimento media di scrittura log
ibmmq_nha_replication_log_write_average_acknowledgement_size_bytes	gauge	Dimensione riconoscimento media di scrittura log
ibmmq_nha_replication_mq_fdc_file_count	gauge	Conteggio file FDC MQ
ibmmq_nha_replication_queue_manager_file_system_free_space_percent	gauge	File system Gestore code - spazio disponibile
ibmmq_nha_replication_queue_manager_file_system_in_use_bytes	gauge	File system Gestore code - byte in uso
ibmmq_nha_replication_synchronous_compressed_log_sent_bytes	counter	Byte di log compressi sincroni inviati
ibmmq_nha_replication_synchronous_log_data_average_compression_time_seconds	gauge	Tempo medio di compressione dei dati di log sincroni
ibmmq_nha_replication_synchronous_log_data_average_decompression_time_seconds	gauge	Tempo medio di decompressione dei dati di log sincroni
ibmmq_nha_replication_synchronous_log_decompressed_bytes	gauge	Byte di log sincroni decompressi
ibmmq_nha_replication_synchronous_log_sent_bytes	counter	Byte log sincrono inviati
ibmmq_nha_replication_synchronous_uncompressed_log_sent_bytes	counter	Byte di log non compressi sincroni inviati
ibmmq_qmgr_commit_total	counter	Conteggio commit

Metrica	Tipo	Descrizione
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
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

Metrica	Tipo	Descrizione
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
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_free_space_percentage	gauge	File system di log - spazio disponibile
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_occupied_by_extents_waiting_to_be_archived_bytes	gauge	Log - occupato da estensioni in attesa di essere archiviate

Metrica	Tipo	Descrizione
ibmmq_qmgr_log_occupied_by_reusable_extents_bytes	gauge	Log - byte occupati dalle estensioni riutilizzabili
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_required_for_media_recovery_bytes	gauge	Log - byte necessari per il ripristino supporti
ibmmq_qmgr_log_sequence_number_disk_total	gauge	Log - numero di sequenza del log scritto su disco
ibmmq_qmgr_log_sequence_number_quorum_total	gauge	Log - numero di sequenza del log del quorum
ibmmq_qmgr_log_slowest_write_since_restart_seconds	gauge	Log - Scrittura più lenta dal riavvio
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
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

Metrica	Tipo	Descrizione
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
ibmmq_qmgr_persistent_message_browse_bytes_total	counter	Esplorazione messaggi permanenti - conteggio byte
ibmmq_qmgr_persistent_message_browse_total	counter	Esplorazione messaggi permanenti - conteggio

Metrica	Tipo	Descrizione
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
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

Metrica	Tipo	Descrizione
ibmmq_qmgr_topic_mqput_mqput1_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

Informazioni correlate

Metriche pubblicate negli argomenti di sistema

OpenShift V 9.4.0 V 9.4.0 Arresto di un gestore code (mq.ibm.com/stop)

Arrestare un gestore code aggiungendo un'annotazione alla definizione del gestore code.

Informazioni su questa attività

I gestori code creati dall'operatore IBM MQ hanno un StatefulSetassociato. Questo StatefulSet dichiara il numero di Pods da distribuire per un determinato tipo di disponibilità del gestore code mediante il campo '.replicas'. Assume il valore di 1 (Istanza singola), 2 (Istanza multipla) o 3 (NativeHA).

Nota: La modifica manuale del valore nel campo '.replicas' impedisce al gestore code di funzionare correttamente.

In alcuni casi, è possibile che si desideri arrestare il gestore code in modo che StatefulSet abbia un conteggio di repliche pari a 0 e non venga distribuito alcun Pods . Esempi di quando si potrebbe voler eseguire questa operazione includono durante la manutenzione o una procedura di backup.

Nota: Poiché non vi sono gestori code Pods distribuiti quando il gestore code viene arrestato, l'utente e le applicazioni non saranno in grado di accedere al gestore code fino a quando non viene riavviato.

Procedura

- Per arrestare il gestore code, aggiungere la seguente annotazione alla definizione del gestore code nella sezione '.metadata.annotations'.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: my-qm
  annotations:
    "mq.ibm.com/stop" : "true"
```

- Per riavviare il gestore code e riportarlo al suo corretto numero di repliche, rimuovere l'annotazione dal gestore code o impostare il suo valore su 'false'.

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

Accedere al cluster utilizzando **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
  ```

MQ Adv. Visualizzazione dello stato dei gestori code della HA nativa

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** 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, mentre quelle delle istanze di replica potrebbero essere obsolete.

V 9.4.2 Per una configurazione Native HA Cross-Region Replication (CRR), le informazioni fornite dipendono anche dal fatto che l'istanza faccia parte di un gruppo Live o Recovery. Le informazioni del gruppo Live sono più definitive, ma possono essere ancora non aggiornate poiché il gruppo Recovery invia solo informazioni periodiche.

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.
- V 9.4.2 Visualizza lo stato dei gruppi a cui l'istanza appartiene o a cui è collegata.

I seguenti campi di stato sono utilizzati per segnalare lo stato di configurazione di Native HA e Native HA Cross-Region Replication (CRR) di un'istanza di queue manager. Questi campi appaiono sempre:

RUOLO
Specifica il ruolo corrente dell'istanza. È uno di `Active`, `Replica`, `Unknown`, (per un gruppo di recupero) `Leader`, o `Not Configured`.

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*.

V 9.4.2 **GRPLSN**
Specifica il numero di sequenza del log (LSN) che è stato replicato tra un quorum di istanze nel gruppo HA, in formato " *nnnnn:nnnnn:nnnnn:nnnnn* ". Vuoto se l'LSN non è noto.

V 9.4.2 **GRPNAME**
Specifica il nome del gruppo HA. Vuoto se non è stato configurato un nome.

V 9.4.2 **GRPROLE**
Specifica il ruolo attuale del gruppo HA ed è uno dei seguenti:

- `Live` - Un gruppo dal vivo
- `Recovery` - Un gruppo di recupero
- `Pending live` - Un gruppo in attesa di diventare `Live`
- `Pending recovery` - Un gruppo in attesa di diventare `Recovery`
- `Not configured` - L'HA nativo non è stato configurato
- `Unknown` - Il ruolo attuale non può essere determinato

Questi campi appaiono quando si richiedono informazioni estese per tutte le istanze utilizzando il parametro `-x`:

REPLADDR
L'indirizzo di replica dell'istanza del gestore code.

COLLEGA
Indica se l'istanza è connessa all'istanza attiva.

BACKLOG
Indica il numero di KB che l'istanza è dietro l'istanza attiva.

CONNETTIN
Indica se l'istanza denominata è connessa a questa istanza.

ALTDATE
Indica la data dell'ultimo aggiornamento di queste informazioni. Vuoto se non è mai stato aggiornato.

ALTTIME
Indica l'ora dell'ultimo aggiornamento di queste informazioni. Vuoto se non è mai stato aggiornato.

V 9.4.2 **ACKLSN**
Indica l'LSN che l'istanza ha riconosciuto come scritto nel registro di recupero, in formato *nnnnn:nnnnn:nnnnn:nnnnn*. Vuoto se il valore LSN non è noto.

V 9.4.2 **Stato HA**

Indica lo stato operativo di questa istanza ed è uno dei seguenti valori:

- `Normal` - l'istanza funziona normalmente.
- `Checking` - l'istanza viene controllata per garantire la coerenza del registro di ripristino.
- `Synchronizing` - il sistema sta inviando dati per sincronizzarlo.
- `Rebasing` - un nuovo registro di ripristino viene inviato all'istanza.
- `Disk full` - l'istanza ha il disco pieno.
- `Disconnected` - non può comunicare con l'istanza.
- `Unknown` - lo stato dell'istanza è sconosciuto.

V 9.4.2 **SYNCTIME**
Indica l'ora in cui questa istanza è stata sincronizzata l'ultima volta con il gestore della coda attivo, in formato ISO 8601. Vuoto se l'ora non è nota.

V 9.4.2 Questi campi appaiono quando si richiedono informazioni sul gruppo utilizzando il parametro `-g`. Questi campi sono utili per visualizzare lo stato quando il gruppo fa parte di una configurazione Native HA CRR:

V 9.4.2 **GRPNAME**
Specifica il nome del gruppo HA. Vuoto se non è stato configurato un nome.

V 9.4.2 **GRPROLE**
Specifica il ruolo attuale del gruppo HA ed è uno dei seguenti:

- `Live` - Un gruppo dal vivo
- `Recovery` - Un gruppo di recupero
- `Pending live` - Un gruppo in attesa di diventare `Live`
- `Pending recovery` - Un gruppo in attesa di diventare `Recovery`
- `Unknown` - Il ruolo attuale non può essere determinato
- `Not configured` - L'HA nativo non è stato configurato

V 9.4.2 **GRPADDR**
L'indirizzo IP utilizzato per connettersi al gruppo. Viene impostato su `Unknown` se l'indirizzo non è noto perché non si è verificata una connessione al gruppo.

V 9.4.2 **GRPVER**
Indica la versione della capogruppo corrente nel formato `V.R.M.F`, dove `V.R.M.F` è il numero di Version, Release, Modification e Fix Pack. Se il valore non è noto, viene impostato a `?.?.?.?`

V 9.4.2 **CONNGRP**
Indica se il gruppo è collegato al gruppo locale ed è uno dei seguenti:

- `yes` - il gruppo è collegato.
- `no` - il gruppo è disconnesso.
- `unknown` - lo stato della connessione è sconosciuto.

- suspended - il gruppo è collegato, ma il gruppo non può replicare i dati finché non viene risolta un'incompatibilità di configurazione tra i gruppi.

Questo campo viene visualizzato solo per i gruppi diversi dal gruppo locale.

V 9.4.2 **INSYNC**

Viene visualizzato solo per un gruppo con il ruolo Recovery o Pending live. Indica se il gruppo può diventare Live senza perdita di dati in caso di failover. (Si noti che un gruppo di Recovery potrebbe non essere molto spesso sincronizzato con il gruppo di Live perché il registro viene replicato in modo asincrono. Utilizzare i valori RCOVLSN e RCOVTIME per indicare se potrebbe verificarsi una perdita di dati in un failover non pianificato).

V 9.4.2 **SYNCTIME**

Viene visualizzato solo per un gruppo con il ruolo Recovery o Pending live. Indica l'ora in cui questo gruppo è stato sincronizzato l'ultima volta con il gruppo dell' Live , in formato ISO 8601 (vuoto se non è noto). (Si noti che un gruppo di Recovery potrebbe non essere molto spesso sincronizzato con il gruppo di Live perché il registro viene replicato in modo asincrono. Utilizzare i valori RCOVLSN e RCOVTIME per indicare se potrebbe verificarsi una perdita di dati in un failover non pianificato).

V 9.4.2 **BACKLOG**

Viene visualizzato solo per un gruppo con il ruolo Recovery o Pending live. Indica il numero di KB che il gruppo è indietro rispetto al gruppo live.

V 9.4.2 **GRSTATO**

Indica lo stato operativo di questo gruppo ed è uno dei seguenti valori:

- Normal - il gruppo funziona normalmente.
- Checking - il gruppo viene controllato per garantire la coerenza del registro di ripristino.
- Synchronizing - il gruppo riceve dati per sincronizzarsi.
- Rebasing - un nuovo registro di ripristino viene inviato all'istanza.
- Waiting for connection - il gruppo è in attesa di una connessione da un gruppo con il ruolo Live o Pending Recovery.
- Partitioned - due gruppi credono di essere il gruppo Live (cioè uno stato di "cervello diviso").
- Unknown - lo stato del gruppo è sconosciuto.

V 9.4.2 **RCOVLSN**

Un numero di sequenza del registro (LSN) che il gruppo potrebbe recuperare, in formato *nnnnn:nnnnn:nnnnn:nnnnn*. Qualsiasi dato scritto nel registro di ripristino dopo questo punto potrebbe andare perso se si verifica un failover non pianificato. Questo valore può essere passato nell'opzione -s del comando **dmpmqlog**.

V 9.4.2 **RCOVTIME**

Un orario a cui il gruppo può recuperare, in formato ISO 8601. Qualsiasi dato scritto nel registro di ripristino dopo questo momento potrebbe andare perso se si verifica un failover non pianificato. Questo valore può essere passato nelle opzioni -t o -u del comando **dmpmqlog**.

V 9.4.2 **INITLSN**

Viene visualizzato solo per un gruppo con il ruolo Live o Pending recovery. Indica il numero di sequenza del registro (LSN) dell'ultimo record di registro recuperato quando il gruppo Native HA è diventato attivo, nel formato *nnnnn:nnnnn:nnnnn:nnnnn*, o vuoto se il gruppo non è attivo.

V 9.4.2 **Tempo ini**

Viene visualizzato solo per un gruppo con il ruolo Live o Pending recovery. Indica l'ora dell'ultimo record di registro recuperato quando il gruppo HA nativo è diventato inizialmente attivo, in formato ISO 8601. Vuoto, se il gruppo non è attivo.

V 9.4.2 **TEMPO DI VITA**
Viene visualizzato solo per un gruppo con il ruolo `Live` o `Pending recovery`. Indica l'ora in cui il gruppo è diventato per la prima volta il gruppo live, in formato ISO 8601. Vuoto, se il gruppo non è attivo.

V 9.4.2 **ALTDATE**
Indica la data dell'ultimo aggiornamento di queste informazioni.

V 9.4.2 **ALTTIME**
Indica l'ora dell'ultimo aggiornamento di queste informazioni.

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)
```

V 9.4.2 Un'istanza di BOB che è leader di un gruppo di recupero in una configurazione CRR Native HA riporta il seguente stato:

```
QMNAME(BOB)                STATUS(Recovery group leader)
```

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) V 9.4.2
GRPLSN(<0:25:365:1000>) GRPNAME() GRPROLE(Live)
```

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) V 9.4.2
GRPLSN(<0:25:365:1000>) GRPNAME() GRPROLE(Live)
```

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) V 9.4.2
GRPLSN() GRPNAME() GRPROLE(Live)
```

V 9.4.2 Un'istanza attiva di un gestore di code chiamato BOB appartenente al gruppo Live in una configurazione CRR Native HA potrebbe riportare il seguente stato:

```
QMNAME(BOB)         ROLE(Active) INSTANCE(inst1) INSYNC(yes) QUORUM(3/3)
GRPLSN(<0:25:365:1000>) GRPNAME(alpha) GRPROLE(Live)
```

V 9.4.2 La stessa istanza, quando il gruppo era in attesa di diventare gruppo di recupero, potrebbe riportare il seguente stato:

```
QMNAME(BOB)         ROLE(Active) INSTANCE(inst1) INSYNC(yes) QUORUM(3/3)
GRPLSN(<0:25:365:1000>) GRPNAME(alpha) GRPROLE(Pending recovery)
```

V 9.4.2 Un gestore di code chiamato CAROL che non è un'istanza di una configurazione Native HA o Native HA CRR riporta il seguente stato:

```
QMNAME(CAROL)       ROLE(Not configured) INSTANCE() INSYNC() QUORUM() GRPLSN() GRPNAME()
GRPROLE(Not configured)
```

- 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
```

Ad esempio, se si emette questo comando sul nodo che esegue l'istanza attiva del gestore di code BOB, dove tutte le istanze sono sincronizzate, si potrebbe ricevere il seguente stato:

```
QMNAME(BOB)         ROLE(Active) INSTANCE(inst1) INSYNC(yes) QUORUM(2/3)
GRPLSN(<0:25:365:1000>) GRPROLE(Live) GRPNAME()
 INSTANCE(inst1) ROLE(Active)  REPLADDR(localhost) CONNACTV(yes) INSYNC(yes)
BACKLOG(0)  CONNINST(yes) ACKLSN(<0:26:365:1000>)  HASTATUS(Normal)
SYNCTIME(2023-12-20T17:13:27.853412Z) ALTDATE(2023-12-20) ALTTIME(17.13.27)
 INSTANCE(inst3) ROLE(Replica) REPLADDR(localhost) CONNACTV(yes) INSYNC(yes)
BACKLOG(0)  CONNINST(yes) ACKLSN (<0:24:365:1000>) HASTATUS(Normal)
SYNCTIME(2023-12-20T17:13:27.853412Z) ALTDATE(2023-12-20) ALTTIME(17.13.27)
 INSTANCE(inst2) ROLE(Replica) REPLADDR(localhost) CONNACTV(yes) INSYNC(yes)
BACKLOG(0)  CONNINST(yes) ACKLSN (<0:25:365:1000>) HASTATUS(Synchronizing)
SYNCTIME(2023-12-20T17:13:22.123321Z) ALTDATE(2023-12-20) ALTTIME(17.13.27)
```

In un altro esempio, se si emette questo comando su un nodo che esegue un'istanza di replica del gestore di code BOB, che è in ritardo, si potrebbe ricevere il seguente stato:

```
QMNAME(BOB)         ROLE(Replica) INSTANCE(inst2) INSYNC(no) QUORUM(2/3)
GRPLSN(<0:25:365:1000>) GRPROLE(Live) GRPNAME()
 INSTANCE(inst1) ROLE(Active)  REPLADDR(localhost) CONNACTV(yes) INSYNC(yes)
BACKLOG(0)  CONNINST(yes) ACKLSN(<0:26:365:1000>)  HASTATUS(Normal)
SYNCTIME(2023-12-20T17:13:27.853412Z) ALTDATE(2023-12-20) ALTTIME(17.13.27)
 INSTANCE(inst3) ROLE(Replica) REPLADDR(localhost) CONNACTV(yes) INSYNC(yes)
BACKLOG(0)  CONNINST(yes) ACKLSN (<0:24:365:1000>) HASTATUS(Normal)
SYNCTIME(2023-12-20T17:13:27.853412Z) ALTDATE(2023-12-20) ALTTIME(17.13.27)
 INSTANCE(inst2) ROLE(Replica) REPLADDR(localhost) CONNACTV(yes) INSYNC(no)
BACKLOG(435) CONNINST(yes) ACKLSN (<0:25:365:1000>) HASTATUS(Synchronizing)
SYNCTIME(2023-12-20T17:13:22.123321Z) ALTDATE(2023-12-20) ALTTIME(17.13.27)
```

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) GRPLSN()
GRPROLE() GRPNAME()
 INSTANCE(inst1) ROLE(Unknown) REPLADDR(9.20.123.45) CONNACTV(Unknown) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ACKLSN()  HASTATUS(Unknown) SYNCTIME() ALTDATE() ALTTIME()
ALTDATE() ALTTIME()
 INSTANCE(inst2) ROLE(Unknown) REPLADDR(9.20.123.46) CONNACTV(Unknown) INSYNC(Unknown)
BACKLOG(Unknown) CONNINST(No) ACKLSN()  HASTATUS(Unknown) SYNCTIME() ALTDATE() ALTTIME()
ALTDATE() ALTTIME()
 INSTANCE(inst3) ROLE(Unknown) REPLADDR(9.20.123.47) CONNACTV(No)      INSYNC(Unknown)
```

```
BACKLOG(Unknown) CONNINST(No) ACKLSN()  HASTATUS(Unknown) SYNCTIME() ALTDATE() ALTTIME()
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)
```

- V 9.4.2

  Per determinare lo stato operativo di tutti i gruppi Native HA in una configurazione Native HA CRR:

  ```
  oc exec -t Pod dspmq -o nativeha -g -m QMgrName
  ```

  Se si impartisce questo comando a un membro del gruppo Live, denominato alpha, si potrebbe ricevere il seguente stato per alpha e il gruppo Recovery, beta:

  ```
  GRPNAME(alpha) GRPROLE(Live) GRPADDR(9.20.135.129) GRPVER(09040200)
  GRSTATUS(Normal) RCOVLSN(<0:0:104:65163>) RCOVTIME(2023-12-20T17:37:27.114787Z)
  INITLSN(<0:0:101:352>) INITTIME(2023-12-19T09:13:27.512294)
  LIVETIME(2023-12-19T06:14:27.512294) ALTDATE(2023-12-20) ALTTIME(17.13.27)
  ```

  ```
  GRPNAME(beta) GRPROLE(Recovery) GRPADDR(9.20.135.130) GRPVER(09040300)
  CONNGRP(yes) GRSTATUS(Synchronizing) RCOVLSN(<0:0:19:16723>)
  RCOVTIME(2023-12-20T17:36:23.463209Z) BACKLOG(5619) INSYNC(no) SYNCTIME()
  ALTDATE(2023-12-20) ALTTIME(17.13.27)
  ```

Riferimenti correlati

comando dspmq (visualizza gestori code)

V 9.4.2 Commutazione e failover nativi HA CRR

Una configurazione Native HA Cross-Region Replication (CRR) consente di spostare l'esecuzione di un gestore di code in un'altra regione.

È possibile utilizzare una configurazione Native HA CRR per fornire una soluzione di disaster recovery. Se si verifica un disastro nel sito attivo, è possibile eseguire un failover e continuare a eseguire il gestore code nel sito di ripristino. In questa situazione, non è possibile garantire che i registri di ripristino siano sincronizzati quando si verifica il failover, quindi potrebbero verificarsi delle perdite di dati. Allo stesso modo, quando il sito guasto viene ripristinato, potresti avere due istanze attive del gestore di code e quindi potresti accumulare alcuni dati partizionati (split-brain).

È possibile utilizzare una configurazione Native HA CRR per garantire la continuità durante la manutenzione o i test. In questa circostanza, il passaggio è controllato in modo tale da non causare alcuna perdita di dati. Allo stesso modo, quando si torna indietro non ci sono dati partizionati da risolvere.

V 9.4.2 Completare una commutazione pianificata su una configurazione Native HA CRR

È possibile eseguire un passaggio pianificato tra i gruppi live e recovery in una configurazione di replica interregionale (CRR) HA nativa.

Informazioni su questa attività

Potrebbe essere necessario passare dal gruppo live al gruppo di ripristino se si desidera testare la configurazione o se è necessario eseguire la manutenzione.

Quando si esegue un passaggio pianificato, il sistema adotta misure per garantire che non si perdano dati o si verifichi un problema di partizionamento (split-brain). È possibile richiedere il passaggio modificando il ruolo di ciascun gruppo. Per un po' entrambi i gruppi entrano in uno stato di sospensione mentre i loro registri di ripristino vengono sincronizzati. Quando ciò viene raggiunto, i gruppi adottano i loro nuovi ruoli, un nuovo leader viene eletto nel nuovo gruppo live e assume il ruolo attivo e gestisce il queue manager.

Procedura

1. Specificare che il gruppo live deve diventare il gruppo di ripristino impostando il suo `GroupRole` su `Recovery` nella sezione `NativeHALocalInstance` del file `qm.ini` .

   Se si utilizza l'operatore " MQ ", l'argomento " `GroupRole` " viene modificato impostando il campo " `.spec.queueManager.availability.nativeHAGroups.local.role` " nel file " QueueManager " YAML. Ad esempio, per modificare il ruolo del gruppo Live in Recovery, impostare il valore del campo `.spec.queueManager.availability.nativeHAGroups.local.role` su `Recovery` per il gruppo attualmente live.

   Per verificare che il passaggio sia avvenuto con successo, ottenere gli stati dei pod utilizzando il comando **`oc get pods`** ; un pod dovrebbe essere in stato di pronto:

   ```
   NAME                 READY   STATUS    RESTARTS   AGE
   exampleqm-ibm-mq-0   1/1     Running   0          2m45s
   exampleqm-ibm-mq-1   0/1     Running   0          2m45s
   exampleqm-ibm-mq-2   0/1     Running   0          2m45s
   ```

   Per verificare lo stato del gruppo di recupero, eseguire il seguente comando:

   ```
   oc exec -t <leader-pod> -- dspmq -o nativeha -g
   ```

   Dove <leader-pod> è il nome del pod pronto 1/1 (exampleqm-ibm-mq-0 nell'esempio):

   Controllare che il GRPROLE sia in attesa di recupero.
2. Specificare che il gruppo di ripristino deve diventare il gruppo live impostando il suo `GroupRole` su `Live` nella sezione `NativeHALocalInstance` del file `qm.ini` .

   Se si utilizza l'operatore " MQ ", l'argomento " `GroupRole` " viene modificato impostando il campo " `.spec.queueManager.availability.nativeHAGroups.local.role` " nel file " QueueManager " YAML. Ad esempio, per modificare il ruolo del gruppo Live in Live, impostare il valore del campo `.spec.queueManager.availability.nativeHAGroups.local.role` su `Live` per il gruppo che è attualmente in ripristino.

   Se si emette un comando **`dspmq -o nativeha -g`** per questo queue manager dopo che ogni istanza nel gruppo è stata riavviata, si vedrà un GRPROLE di `Pending live`.

   Dopo il riavvio di ogni istanza del gruppo, eseguire il seguente comando:

   ```
   oc exec -t <active-pod> -- dspmq -o nativeha -g
   ```

   Dove <active-pod> è il nome del pod 1/1 pronto visualizzato dal comando `oc get pods` . GRPROLE è `Pending live` seguito da `live` dopo la sincronizzazione dei gruppi.
3. Quando i registri di ripristino sul sito live e di ripristino sono sincronizzati, il gruppo di ripristino diventa live e elegge un leader come istanza di gestione della coda attiva. Da qui, le app si ricollegheranno automaticamente a questa istanza e qualsiasi carico di lavoro riprenderà normalmente.

Operazioni successive

Una volta completata la manutenzione, è possibile tornare al gruppo live originale ripetendo la procedura.

V 9.4.2 Completare un failover non pianificato su una configurazione Native HA CRR

Si esegue un failover non pianificato tra i gruppi live e di ripristino in una configurazione di replica interregionale (CRR) ad alta disponibilità nativa quando si verifica un disastro nel sito live e questo non può più funzionare.

Informazioni su questa attività

Quando il gruppo live si spegne, il gruppo di ripristino non può rilevarlo poiché semplicemente smette di ricevere dati di log replicati, il che potrebbe semplicemente significare che non c'è un carico di lavoro in

corso. Un failover non pianificato deve quindi essere eseguito manualmente cambiando il ruolo del gruppo di ripristino in live e notificandogli che il gruppo precedentemente live è disabilitato in modo che non tenti di replicare i dati su di esso.

Il gruppo di ripristino potrebbe non essere sincronizzato con il gruppo live poiché i dati di log vengono replicati in modo asincrono. Se il gruppo di ripristino non è sincronizzato con il gruppo live, i dati vengono persi quando si verifica un failover non pianificato. Prima di avviare un failover non pianificato, inviare il comando " **dspmq -o nativeha -g** " sul gruppo di ripristino e utilizzare i valori "RCOVLSN" e "RCOVTIME" per visualizzare il numero di sequenza del registro (LSN) e l'ora a cui il gruppo può ripristinarsi.

Procedura

1. Assicurarsi che le applicazioni siano state scollegate dal gruppo attivo e siano configurate per ricollegarsi al gruppo che è attualmente il gruppo di ripristino.
2. Specificare che il gruppo di ripristino deve diventare il gruppo attivo completando i seguenti passaggi:
   a. Imposta il suo " `GroupRole` " su " `Live` " nella sezione " `NativeHALocalInstance` " del file " `qm.ini` ". Ad esempio, se si utilizza l'operatore " MQ ", modificare l'impostazione di " `.spec.queueManager.availability.nativeHAGroups.local.role` " in " `Live`".
   b. Impostare il campo " `Enabled` " su " `False` " nella sezione " `NativeHARecoveryGroup` " del file " `qm.ini` ". Ad esempio, se si utilizza l'operatore " MQ ", modificare l'impostazione di " `.spec.queueManager.availability.nativeHAGroups.remotes.enabled` " in " `false`".

   Se si emette un comando **dspmq -o nativeha -g** per questo queue manager dopo che ogni istanza nel gruppo è stata riavviata, si vedrà un GRPROLE di `Live`. Si tiene un'elezione tra i membri del gruppo e uno di loro diventa l'istanza attiva e gestisce il lavoro del gestore della coda.
3. Per riavviare la replica dopo il failover, è possibile utilizzare il gruppo live originale come gruppo di ripristino oppure creare un nuovo gruppo di ripristino in un nuovo sito.
   - Se il sito live originale viene ripristinato, prima di riavviare il gruppo live originale, modificare il ruolo del gruppo in ripristino. In questo modo si garantisce che il gruppo non riprenda il suo ruolo attivo.
   - Se il sito live originale non può essere ripristinato, creare un nuovo gruppo di ripristino Native HA in un nuovo sito.

   Per configurare il gruppo di ripristino, completare i passaggi descritti in “Ripristino della replica dopo un failover non pianificato” a pagina 237.
4. Se la configurazione sul sito live originale non viene modificata e il gruppo riprende il suo ruolo live quando viene ripristinato, si avranno due istanze attive del gestore code. In tal caso, viene rilevato lo stato di partizione e non avviene più alcuna replica tra i due gruppi. Devi decidere quali dati conservare prima di ripristinare i ruoli originali di accesso e ripristino. Vedere “Risolvere un problema di partizionamento (split brain)” a pagina 238 per indicazioni.

Informazioni correlate

dspmq (display queue managers)

V 9.4.2 Ripristino della replica dopo un failover non pianificato

Ripristino della replica dal gruppo live a un gruppo di ripristino dopo un failover non pianificato.

Informazioni su questa attività

Dopo un failover non pianificato, se il sito live originale viene ripristinato, può essere riconfigurato per diventare il gruppo di ripristino. Per evitare che il gestore della coda entri in uno stato di partizionamento ("split-brain"), completare questa operazione prima di riavviare il gruppo Native HA sul sito live originale.

È inoltre possibile completare questi passaggi quando si crea un nuovo gruppo di ripristino dopo un failover non pianificato, se il sito live originale non può essere ripristinato.

Procedura

1. Specificare che il gruppo deve diventare il gruppo di ripristino completando i seguenti passaggi:
   a) Impostare l' GroupRole e su Recovery nella sezione " NativeHALocalInstance " del file " `qm.ini` ". Ad esempio, se si utilizza l'operatore " MQ ", modificare l'impostazione del campo " `.spec.queueManager.availability.nativeHAGroups.local.role` " nel file " QueueManager " da "YAML" a "Recovery".
   b) Impostare il campo Enabled su False nella sezione " NativeHARecoveryGroup " del file " qm.ini ". Ad esempio, se si utilizza l'operatore " MQ ", modificare l'impostazione del campo " `.spec.queueManager.availability.nativeHAGroups.remotes.enabled` " nel file " QueueManager " YAML su "false".

      Se si emette un comando **`dspmq -o nativeha -g`** su un'istanza del gruppo dopo che ogni istanza è stata riavviata, si vedrà un GRPROLE di `Recovery`. Si tiene un'elezione tra i membri del gruppo e uno di loro diventa il leader del gruppo di recupero.
2. Riavvia la replica dal gruppo live al gruppo di ripristino impostando il campo `Enabled` su `True` nella sezione NativeHARecoveryGroup del file `qm.ini` del gruppo live. Ad esempio, se si utilizza l'operatore " MQ ", modificare l'impostazione del campo " `.spec.queueManager.availability.nativeHAGroups.remotes.enabled` " nel file " QueueManager " YAML su "true".

V 9.4.2 Risolvere un problema di partizionamento (split brain)

Una condizione di partizionamento, o "split-brain", può verificarsi se un gestore di code in una configurazione di replica interregionale (CRR) nativa HA viene eseguito contemporaneamente in entrambi i gruppi HA.

La condizione può verificarsi quando si verifica un failover non pianificato. La comunicazione con il gruppo Live è interrotta, quindi imposti il gestore della coda come principale sul gruppo Recovery e questo inizia a funzionare. Nel frattempo, il gestore della coda è ancora in esecuzione sul gruppo Live originale. È possibile che il vecchio gruppo Live sia ancora in grado di aggiornare il proprio registro e questo crea un ramo irrisolvibile nei dati gestiti da entrambi i due gruppi. È necessario eliminare i dati da uno dei gruppi prima che possano essere riuniti.

Per risolvere la situazione di partizionamento, è necessario scegliere quali dati di gruppo conservare e quali eliminare. Potrebbe essere che i gruppi non fossero sincronizzati quando è stato attivato il failover non pianificato, oppure che il carico di lavoro dell'applicazione sia stato in grado di essere elaborato da entrambi i gruppi. La definizione di dati "migliori" è in gran parte soggettiva. Ad esempio, mentre un gruppo potrebbe aver memorizzato più dati nel proprio registro, questi potrebbero essere in gran parte immagini multimediali registrate automaticamente di oggetti di v IBM MQ , mentre l'altro gruppo potrebbe aver elaborato transazioni commerciali più preziose. Qualunque sia il gruppo scelto, è probabile che, nel recupero dalla condizione di partizionamento, il messaggio non replicato e gli esiti della transazione sul gruppo da eliminare saranno utili per riconciliare l'eventuale perdita di dati.

IBM MQ rileva una condizione di partizionamento quando un gruppo con un ruolo Live si connette a un altro gruppo che sta eseguendo un ruolo Live. Se si invia il comando `dspmq -o nativeha -g` in entrambi i gruppi, viene restituito lo stato `GRSTATUS(Partitioned)` . La replica dei dati di registro è sospesa tra entrambi i gruppi. Lo stato di " `Partitioned` " continua a essere segnalato fino a quando il gruppo non è in grado di replicare dati di registro sufficienti a un altro gruppo con un ruolo di "Recovery".

Se non è ovvio quale gruppo mantenere e se entrambi i gruppi hanno ancora spazio disponibile al momento del failover non pianificato, è possibile confrontare i contenuti dei log.

Utilizzare il comando " **`dspmq -o nativeha -g`** " per identificare l'LSN (numero di sequenza del registro) del punto esatto nel registro in cui un gruppo è diventato attivo. LSN viene restituito nella forma `INITLSN (`*`nnnnn`*`:`*`nnnnn`*`:`*`nnnnn`*`:`*`nnnnn`*`)`. L'ora di questo evento è riportata anche dall'attributo " `INITTIME` ". Quindi utilizzare il comando " **`dmpmqlog -s`** *`startLSN`* " per esaminare il contenuto di un registro da quel punto. Utilizzare **`dspmq`** e **`dmpmqlog`** in combinazione per identificare i record di log che sono stati scritti da un gruppo e non replicati nell'altro. Il comando " **`dmpmqlog`** " riporta anche un riepilogo dell'attività di messaggistica e transazione, che può essere utile al momento della scelta.

Esempio

I gruppi "alfa" e "beta" sono stati inizialmente configurati in una configurazione Live and Recovery. È stata presa la decisione di convertire il gruppo "beta" da Recovery a Live e si è verificato un failover non pianificato. Tuttavia, il gruppo "alfa" è riuscito a portare avanti alcuni lavori di applicazione che non sono stati replicati nel gruppo "beta". Nel frattempo, alcune applicazioni collegate alla "beta" hanno inviato alcuni messaggi. L'INITLSN del gruppo "beta" indica il punto nel registro in cui il gruppo "beta" si è separato dal contenuto del gruppo "alfa".

Per il processo di confronto è necessaria una copia dei dati del gestore di coda e il gestore di coda deve essere inattivo. Ciò può essere ottenuto in un ambiente di tipo " Kubernetes " riducendo a zero la dimensione del set stateful del queue manager o eseguendo una copia dei dati, ad esempio tramite un'istantanea del volume persistente.

Dopo aver completato l'analisi dei dati di log e aver scelto quali dati di gruppo scartare, è possibile scartarli modificando il ruolo del gruppo in Ripristino ed eliminando tutti i volumi persistenti del gestore di coda o rimuovendo le informazioni del gruppo Ripristino.

Informazioni correlate

dspmq

dmpmqlog

OpenShift MQ Adv. Conclusione dei gestori di coda HA nativi

Per IBM MQ in container, è possibile utilizzare il comando **endmqm** per terminare un gestore di code attivo o una replica che fa parte di un gruppo Native HA.

Prima di iniziare

Nota: Queste informazioni si applicano solo agli ambienti container.

Informazioni su questa attività

La procedura di arresto di un gestore di code che fa parte di un gruppo Native HA dipende dal fatto che si tratti di un'istanza attiva o di una replica. Quando si termina uno dei due tipi di istanza, viene effettuato un controllo per garantire che la terminazione dell'istanza non interrompa il quorum del gruppo Native HA. Se il quorum viene meno, il comando **endmqm** fallisce.

Quando si impartisce un comando **endmqm**, le altre istanze del gruppo vengono avvertite che ciò sta accadendo, in modo che non segnalino errori quando la connessione si interrompe.

Se un'istanza attiva perde il quorum a causa dell'interruzione o della disconnessione di troppe istanze di replica, l'istanza attiva attende per un periodo di tempo configurabile prima di terminare completamente. In questo modo si ottiene un periodo di tempo per chiudere l'elaborazione con grazia, anziché interrompere le connessioni delle applicazioni. Questo valore di timeout può essere specificato dall'attributo `QuorumConnectivityTimeout` nella stanza `NativeHALocalInstance` del file `qm.ini`. Il valore predefinito è 0 secondi.

Procedura

- Per terminare l'istanza attiva di un gestore di code, eseguire il seguente comando sul nodo in cui è in esecuzione l'istanza attiva:

  ```
  endmqm -s QMgrName
  ```

  - Specificare l'opzione `-r` per aiutare le applicazioni client a riconnettersi a un'altra istanza.
  - Se l'istanza non è quella attiva nel gruppo Native HA, il comando fallisce.
  - Se l'interruzione di questa istanza attiva causerebbe il fallimento del quorum del gruppo, il comando fallisce. (Se altre istanze terminano o diventano non disponibili nello stesso momento in cui si

esegue questo comando, il controllo del quorum potrebbe non rilevarlo, il gruppo Native HA termina e può essere riavviato solo quando sono disponibili abbastanza istanze)

Quando il gestore di code attivo termina, una delle istanze di replica assume il ruolo attivo. Non è possibile specificare quale replica subentra, questo è determinato dalla negoziazione all'interno del gruppo e dipende da quale ha i registri delle transazioni più aggiornati.

- Per terminare un'istanza di replica di un gestore di code, eseguire il seguente comando:

```
endmqm -x QMgrName
```

  - Se l'istanza è quella attiva, il comando fallisce.
  - Se l'interruzione di questa istanza di replica causerebbe il fallimento del quorum del gruppo, il comando fallisce. (Se altre istanze terminano o diventano non disponibili nello stesso momento in cui si esegue questo comando, il controllo del quorum potrebbe non rilevarlo, il gruppo Native HA termina e può essere riavviato solo quando sono disponibili abbastanza istanze)

  Nota: È inoltre possibile utilizzare gli switch -c, -i, -p o -w con il comando **endmqm** sulle istanze Native HA, indipendentemente dal ruolo ricoperto. L'istanza del gestore di code termina, ignorando l'effetto che ha sul quorum del gruppo. Le informazioni vengono comunque condivise con le altre istanze del gruppo. È possibile utilizzare questi interruttori insieme a -s per l'istanza attiva. Non è possibile utilizzare questi interruttori insieme all'interruttore -x per le istanze di replica.

Risoluzione dei problemi di IBM MQ nei contenitori

Se stai avendo problemi con l'esecuzione di IBM MQ in un contenitore, puoi utilizzare le tecniche qui descritte per aiutarti a diagnosticare e risolvere i problemi.

Procedura

- “Risoluzione dei problemi di riavvii non pianificati di IBM MQ nei contenitori” a pagina 240.
- “Risoluzione dei problemi con IBM MQ Operator” a pagina 242.

OpenShift CP4I Kubernetes Risoluzione dei problemi di riavvii non pianificati di IBM MQ nei contenitori

Nella maggior parte dei sistemi di gestione dei contenitori come Red Hat OpenShift Container Platform e Kubernetes, i contenitori vengono comunemente riavviati. Non è normale che un contenitore sia di lunga durata. Questo argomento illustra il ciclo di vita del contenitore, il modo in cui è possibile esaminare un riavvio e le cause di un riavvio del contenitore non pianificato.

Se non sono stati riscontrati problemi con la propria distribuzione IBM MQ e questa continua ad essere eseguita come previsto, è probabile che la soluzione stia funzionando come previsto. È possibile che nel log del contenitore venga visualizzato un messaggio simile al seguente:

```
Signal received: terminated
```

Ciò significa che il segnale SIGTERM è stato inviato al contenitore MQ, chiedendogli di terminare. Linux i contenitori hanno la responsabilità di rispondere ai segnali di POSIX, che sono messaggi standardizzati inviati a un programma per attivare un comportamento. Il segnale SIGTERM è usato da Kubernetes (e quindi da Red Hat OpenShift Container Platform) per terminare con grazia un contenitore.

Quando il contenitore IBM MQ riceve un segnale SIGTERM, emette un comando `endmqm -w -r -tp` per arrestare il gestore code. Una volta arrestato il gestore code, il contenitore verrà arrestato. Se l'arresto del gestore code richiede molto tempo, è possibile che venga inviato un segnale SIGKILL, che terminerà immediatamente i processi Linux . La quantità di tempo tra SIGTERM e SIGKILL è nota come "periodo di tolleranza di terminazione" in Kubernetesed è configurabile sulla risorsa `QueueManager` (se si utilizza IBM MQ Operator) o direttamente sulla risorsa Pod. Il valore predefinito è 30 secondi, di cui un secondo è riservato per l'arresto del contenitore e il resto viene assegnato a IBM MQ. Ad esempio, nel caso

predefinito, viene emesso un `endmqm -w -tp 29` , che indica al gestore code di avere 29 secondi per l'arresto.

Motivi della chiusura del container

Kubernetes può riavviare un contenitore all'interno di un Pod esistente, oppure può eliminare il Pod. Ci sono campi nella sezione "status" del Pod che aiutano a indicare se si tratta di un riavvio del contenitore, ad esempio:

```
status:
  containerStatuses:
  - restartCount: 2
    started: true
    state:
      running:
        startedAt: "2025-01-20T12:03:15Z"
  startTime: "2024-12-28T13:46:12Z"
```

In questo esempio, si può notare che il contenitore è stato riavviato e che il contenitore è molto più recente del Pod. Questo indica che un singolo contenitore si è riavviato, il che probabilmente significa che il contenitore non è riuscito a eseguire il probe di liveness. In Kubernetes è possibile configurare una sonda di liveness per verificare che un contenitore sia ancora sano. IBM MQ Operator imposta un probe di attività del gestore code che richiama il comando **dspmq** per verificare uno stato di esecuzione valido. Se il gestore code non è in uno stato di integrità o se l'esecuzione del probe richiede troppo tempo, il kubelet considererà questo un errore. Le soglie per il numero di errori da tollerare sono configurabili, sulla risorsa `QueueManager` (se si utilizza IBM MQ Operator) o direttamente sulla risorsa Pod.

Se non si vedono riavvii del contenitore, probabilmente l'intero Pod è stato sostituito. Vedere Terminazione dei Pod nella documentazione di Kubernetes. Kubernetes utilizza i termini "Pod Disruption" e "eviction" per il processo con cui i Pod sui nodi vengono terminati volontariamente o involontariamente. Ci sono molte ragioni per cui un pod potrebbe essere sfrattato, tra cui:

- **Riavvio continuo**. Il controller IBM MQ Operator o Stateful Set può attivare un aggiornamento continuo dei Pod del gestore di code, se viene apportata una modifica alla configurazione che richiede un rollout. Ad esempio, la modifica della versione dell'immagine, dei file montati, delle variabili d'ambiente, delle annotazioni, delle etichette e altro ancora.
- **Risoluzione da kubelet**. Ciò può essere dovuto a una serie di motivi, tra cui:
  - Il pod può essere terminato perché il Nodo è in fase di arresto (forse come parte di un aggiornamento cluster continuo)
  - Il pod può essere terminato a causa della "pressione" del nodo (dove il kubelet termina proattivamente i pod per recuperare le risorse su un nodo). L'amministratore del cluster Kubernetes può configurare le soglie di eliminazione che potrebbero variare tra i cluster.
- **Prevenzione mediante lo scheduler Kubernetes**. Ciò può verificarsi se il programma di pianificazione Kubernetes deve eseguire un pod con priorità più elevata
- **Nodo tainted** Un nodo può essere "contaminato" e i pod che non tollerano il taint vengono sfrattati. I taint vengono utilizzati dagli amministratori Kubernetes per "respingere" i pod da nodi specifici. Ad esempio, per dire che i pod IBM MQ non devono più essere eseguiti su nodi che hanno hardware speciale che ora è riservato ad altri carichi di lavoro.
- **Richiedi tramite l'API di eliminazione**. Questo può essere richiamato da un amministratore per eliminare i pod
- **Pod garbage collection**. Ciò può verificarsi se il Nodo non è più in servizio o viene rimosso tramite l'API Kubernetes .

Determinazione del motivo per cui un pod del gestore code è stato eliminato

Le potenziali fonti di informazioni che aiutano a capire perché un pod è stato sfrattato includono:

- **Eventi cluster**. Ad esempio, Visualizzazione delle informazioni sull'evento di sistema in un cluster OpenShift Container Platform .

- **Eventi di verifica cluster**. Consultare Visualizzazione dei log di controllo in Red Hat OpenShift Container Platform.
- **Nodi sotto pressione** Ricercare i nodi sotto la pressione della CPU, della rete o della memoria. È possibile visualizzarlo nello stato Nodo. Si noti che nel momento in cui si arriva a guardare, il nodo potrebbe non essere più sotto pressione.
- **Red Hat OpenShift Container Platform Monitoraggio** o altre metriche di monitoraggio potrebbero essere in grado di visualizzare elementi come problemi di latenza del disco. Un'utile metrica Prometheus è `ibmmq_qmgr_log_write_latency_seconds`. Queste informazioni provengono dagli argomenti relativi alle statistiche di MQ .

Informazioni correlate

Documentazione Kubernetes sulla pianificazione, la prevenzione e l'eliminazione

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

- “Raccolta delle informazioni per la risoluzione dei problemi per i gestori code distribuiti con IBM MQ Operator” a pagina 242
- “Risoluzione dei problemi: accesso ai dati del gestore code” a pagina 244

OpenShift CP4I Raccolta delle informazioni per la risoluzione dei problemi per i gestori code distribuiti con IBM MQ Operator

Raccolta delle informazioni per la risoluzione dei problemi che devono essere fornite al supporto IBM quando si genera un nuovo caso di supporto.

Procedura

1. Raccogliere le informazioni sul provider cloud.

   Questo è il provider cloud che ospita il tuo cluster Red Hat OpenShift (ad esempio, IBM Cloud).
2. Raccogliere le informazioni sull'architettura.

   L'architettura del cluster Red Hat OpenShift è una delle seguenti:

   - Linux for x86-64
   - Linux on Power Systems (ppc64le)
   - Linux on IBM Z
3. Raccogliere le informazioni di distribuzione IBM MQ .
   a) Accedi al tuo cluster Red Hat OpenShift , utilizzando una shell `bash/zsh` .
   b) Impostare le seguenti variabili di ambiente:

   ```
   export QM=QueueManager_name
   export QM_NAMESPACE=QueueManager_namespace
   export MQ_OPERATOR_NAMESPACE=mq_operator_namespace
   ```

   Dove *QueueManager_name* è il nome della risorsa QueueManager (`metadata.name` nello YAML del gestore di code), *QueueManager_namespace* è lo spazio dei nomi in cui è distribuita (`metadata.namespace` nello YAML del gestore di code) e *mq_operator_namespace* è lo spazio dei nomi in cui è distribuita IBM MQ Operator . Potrebbe essere lo stesso spazio dei nomi QueueManager.
   c) Immettere i seguenti comandi e fornire tutti i file di output risultanti al supporto IBM .

```
# OCP / Kubernetes: Version
oc version -o yaml > ocversion.yaml

# QueueManager: YAML
oc get qmgr $QM -n $QM_NAMESPACE -o yaml > "queue-manager-$QM.yaml"

# MQ Queue Manager: Pods
oc get pods -n $QM_NAMESPACE -o wide --selector "app.kubernetes.io/instance=$QM" > "qm-
pods-$QM.txt"

# MQ Queue Manager: Pod YAML
oc get pods -n $QM_NAMESPACE -o yaml --selector "app.kubernetes.io/instance=$QM" > "qm-
pods-$QM.yaml"

# MQ Queue Manager: Pod Logs
for p in $(oc get pods -n $QM_NAMESPACE --no-headers --selector "app.kubernetes.io/
instance=$QM" | cut -d ' ' -f 1); do oc logs -n $QM_NAMESPACE --previous "$p" > "qm-logs-
previous-$p.txt"; oc logs -n $QM_NAMESPACE $p > "qm-logs-$p.txt";done

# MQ Queue Manager: Describe Pods
for p in $(oc get pods -n $QM_NAMESPACE --no-headers --selector "app.kubernetes.io/
instance=$QM" | cut -d ' ' -f 1); do oc describe pod $p -n $QM_NAMESPACE > "qm-pod-
describe-$p.txt"; done

# MQ Web UI: Console Log
for p in $(oc get pods -n $QM_NAMESPACE --no-headers --selector "app.kubernetes.io/
instance=$QM" | cut -d ' ' -f 1); do oc cp -n $QM_NAMESPACE --retries=10 "$p:var/mqm/web/
installations/Installation1/servers/mqweb/logs/console.log" "web-$p-console.log"; done

# MQ Web UI: Messages Log
for p in $(oc get pods -n $QM_NAMESPACE --no-headers --selector "app.kubernetes.io/
instance=$QM" | cut -d ' ' -f 1); do oc cp -n $QM_NAMESPACE --retries=10 "$p:var/mqm/web/
installations/Installation1/servers/mqweb/logs/messages.log" "web-$p-messages.log"; done

# MQ Queue Manager: routes defined by operator
oc get routes -n $QM_NAMESPACE -o yaml --selector "app.kubernetes.io/instance=$QM" > "qm-
routes-$QM.yaml"

# MQ Queue Manager: routes to QM
oc get routes -n $QM_NAMESPACE -o yaml --field-selector "spec.to.name=$QM-ibm-mq" > "qm-
routes2-$QM.yaml"

# MQ Queue Manager: stateful set
oc get statefulset -n $QM_NAMESPACE -o yaml ${QM}-ibm-mq > "qm-statefulset-$QM.yaml"

# MQ Queue Manager: revisions of the stateful set
oc get controllerrevisions.apps -o yaml -n $QM_NAMESPACE --selector "app.kubernetes.io/
instance=$QM" > "qm-statefulset-revisions-$QM.yaml"

# MQ Queue Manager: Pod events
for p in $(oc get pods -n $QM_NAMESPACE --no-headers --selector "app.kubernetes.io/
instance=$QM" | cut -d ' ' -f 1); do oc get -o custom-columns="LAST
SEEN:.lastTimestamp,TYPE:.type,REASON:.reason,KIND:.involvedObject.kind,NAME:.involvedObje
ct.name,MESSAGE:.message" event -n $QM_NAMESPACE --field-selector
involvedObject.name="$p" > "qm-pod-events-$p.txt"; done

# MQ Queue Manager: StatefulSet events
oc get events -n $QM_NAMESPACE -o custom-columns="LAST
SEEN:.lastTimestamp,TYPE:.type,REASON:.reason,KIND:.involvedObject.kind,NAME:.involvedObje
ct.name,MESSAGE:.message" --field-selector involvedObject.name="${QM}-ibm-mq" > "qm-
statefulset-events-$QM.txt"

# MQ Queue Manager: services
oc get services -n $QM_NAMESPACE -o yaml --selector "app.kubernetes.io/instance=$QM" >
"qm-services-$QM.yaml"

# MQ Queue Manager: PVCs
oc get pvc -n $QM_NAMESPACE -o yaml --selector "app.kubernetes.io/instance=$QM" > "qm-
pvcs-$QM.yaml"

# MQ Operator: Version
oc get csv -n $QM_NAMESPACE | grep "^ibm-mq\|NAME" > mq-operator-csv.txt

# Cloud Pak Foundational Services: Version
oc get csv -n $QM_NAMESPACE | grep "^ibm-common-service-operator\|NAME" > common-services-
csv.txt

# Cloud Pak for Integration: Version (if applicable)
oc get csv -n $QM_NAMESPACE | grep "^ibm-integration-platform-navigator\|NAME" > cp4i-
csv.txt
```

```
# Output from runmqras (this may take a while to execute)
for p in $(oc get pods -n $QM_NAMESPACE --no-headers --selector "app.kubernetes.io/
instance=$QM" | cut -d ' ' -f 1); do timestamp=$(TZ=UTC date +"%Y%m%d_%H%M%S"); oc exec
-n $QM_NAMESPACE $p -- runmqras -workdirectory "/tmp/runmqras_$timestamp" -section
logger,mqweb,nativeha,trace; oc cp -n $QM_NAMESPACE --retries=10 "$p:tmp/
runmqras_$timestamp/" .; done

# MQ Operator: Pod Log
oc logs -n $MQ_OPERATOR_NAMESPACE $(oc get pods -n $MQ_OPERATOR_NAMESPACE --no-headers --
selector app.kubernetes.io/name=ibm-mq,app.kubernetes.io/managed-by=olm | cut -d ' ' -f
1) > mq-operator-log.txt
```

Note:

- La maggior parte di questi comandi richiede l'accesso allo spazio dei nomi in cui viene distribuito il gestore code. Tuttavia, la raccolta del log IBM MQ Operator potrebbe richiedere anche l'accesso **amministratore del cluster** se IBM MQ Operator è installato **nell'ambito del cluster**.
- Se si ha un gestore di code in esecuzione, assicurarsi che i file di output prodotti da questi comandi non siano vuoti (tranne i log del pod `qm-logs-previous-`*`pod_name`*`.txt` , che dovrebbero essere vuoti se il contenitore non è stato riavviato).

Attività correlate

Raccolta di informazioni per la risoluzione dei problemi

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) Elenca le directory PVC immettendo il seguente comando:

   ```
   ls
   ```

   c) Visualizzare un elenco delle PVC immettendo 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 Troubleshooting container clusters with the OpenShift Monitoring CLI

As a troubleshooting aid, you can query the available container cluster metrics using the OpenShift Monitoring CLI.

Before you begin

OpenShift Monitoring provides a web console and a CLI. For day-to-day operations you might prefer to use the console, as described in “Monitoring container clusters with the OpenShift Monitoring console” on page 218. For gathering troubleshooting information you might prefer to use the CLI, as described in the rest of this task.

About this task

OpenShift Container Platform includes a pre-configured, pre-installed and self-updating monitoring stack that uses an open source Prometheus server. Queue managers running under an IBM MQ Operator can produce metrics compatible with Prometheus. For a complete list of the metrics available for IBM MQ, see “Metriche pubblicate dall' IBM MQ ” on page 220.

OpenShift Monitoring is the default cluster monitoring method. It is installed with the cluster. To query the available container cluster metrics using the OpenShift Monitoring CLI, complete the following steps.

Procedure

1. Enable OpenShift Monitoring for the operands deployed in an IBM MQ container cluster.
   a) Log in to the cluster as a cluster administrator.
   b) Add "`enableUserWorkload: true`" to the configMap `cluster-monitoring-config` in the `openshift-monitoring` namespace. To do this, follow the steps in Enabling monitoring for user-defined projects in the Red Hat OpenShift documentation. If the `cluster-monitoring-config` ConfigMap does not exist in the `openshift-monitoring` namespace, follow the instructions to create it.
2. Install the OpenShift command line tool (oc).

See Getting started with the OpenShift CLI.

3. Query the IBM MQ metrics using the OpenShift Monitoring CLI.

   When using the CLI, first declare the variables by using the **export** command. For example:

```
export OCP_LOGIN_TOKEN=$(oc whoami -t)
export THANOS_QUERIER_HOST=$(oc -n openshift-monitoring get route thanos-querier -o
jsonpath={.spec.host})
export QM_NAMESPACE=namespace-in-which-the-target-queue-manager-is-installed
export QM_METRIC_NAME=IBM-MQ-metric
```

   To run PromQL queries from the CLI, use the **PROMQL_QUERY** command. For example:

   - Get a list of queue managers that have a positive FDC count:

```
export QM_METRIC_NAME=ibmmq_qmgr_fdc_files
export PROMQL_QUERY="$QM_METRIC_NAME{namespace=\"$QM_NAMESPACE\"} > 0"
```

   - Get data for a specific time period:

```
export START=start_time
export END=end_time
export STEP=step_interval
curl -G -s -k -H "Authorization: Bearer $OCP_LOGIN_TOKEN" --data-urlencode
"query=$PROMQL_QUERY" \
--data-urlencode "start=$START" --data-urlencode "end=$END" \
--data-urlencode "step=$STEP" "https://$THANOS_QUERIER_HOST/api/v1/query_range"
```

     where *start_time* and *end_time* specify the time in ISO 8601 format, and *step_interval* is (for example) 15m, or 30s, or 1h.

   You can also export the data to a CSV file:

   a. The following command uses the same headers that display in the web console output shown in the screen capture in “Monitoring container clusters with the OpenShift Monitoring console” on page 218. To export to a CSV file, and pass the same headers that display in the web console output, first download and install jq.

   b. Run the following command:

```
(
echo
"metric_name,container,endpoint,instance,job,namespace,pod,prometheus,qmgr,service,timesta
mp,value"
curl -G -s -k -H "Authorization: Bearer $OCP_LOGIN_TOKEN" --data-urlencode
"query=$PROMQL_QUERY" \
--data-urlencode "start=$START" --data-urlencode "end=$END" --data-urlencode "step=$STEP"
\
"https://$THANOS_QUERIER_HOST/api/v1/query_range"| jq -r '.data.result[] | .metric as
$metric | .values[] | \
[$metric.__name__,
$metric.container,
$metric.endpoint,
$metric.instance,
$metric.job,
$metric.namespace,
$metric.pod,
$metric.prometheus,
$metric.qmgr,
$metric.service,
(.[0] | strftime("%Y-%m-%dT%H:%M:%SZ")), /* converts the timestamp from milliseconds to
ISO 8601 format */
.[1] ] | @csv'
) > your_filename.csv
```

CLI example: Write latency of Queue manager pods over the last 5 hours with a resolution of 10 minutes

```
export QM_METRIC_NAME=ibmmq_qmgr_log_write_latency_seconds
export PROMQL_QUERY="$QM_METRIC_NAME{namespace=\"$QM_NAMESPACE\"} [5h:10m]"
```

Export the data to a CSV file:

```
(
echo
"metric_name,container,endpoint,instance,job,namespace,pod,prometheus,qmgr,service,timestamp,val
ue"
curl -G -s -k -H "Authorization: Bearer $OCP_LOGIN_TOKEN" \
--data-urlencode "query=$PROMQL_QUERY" \
"https://$THANOS_QUERIER_HOST/api/v1/query" | jq -r '.data.result[] | .metric as $metric
| .values[] | \
[$metric.__name__,
$metric.container,
$metric.endpoint,
$metric.instance,
$metric.job,
$metric.namespace,
$metric.pod,
$metric.prometheus,
$metric.qmgr,
$metric.service,
(.[0] | strftime("%Y-%m-%dT%H:%M:%SZ")), /* converts the timestamp from milliseconds to ISO
8601 format */
.[1] ] | @csv'
) > your_filename.csv
```

Related tasks

“Monitoraggio quando si utilizza IBM MQ Operator” on page 217
I gestori code gestiti da IBM MQ Operator possono produrre metriche compatibili con Prometheus. Le metriche del gestore code sono abilitate e servite tramite HTTP per impostazione predefinita.

“Monitoring container clusters with the OpenShift Monitoring console” on page 218
You can query the available container cluster metrics using the OpenShift Monitoring web console.

Related reference

“Metriche pubblicate dall' IBM MQ ” on page 220
I contenitori dei gestori code possono pubblicare metriche compatibili con Red Hat OpenShift Monitoring.

V 9.4.1 Kubernetes Raccolta di informazioni sulla risoluzione dei problemi per i gestori di coda distribuiti su Kubernetes

Raccogliere le informazioni per la risoluzione dei problemi da fornire al IBM Supporto quando si solleva un nuovo caso di supporto.

Procedura

1. Raccogliere le informazioni sul cloud provider.

   Si tratta del provider cloud che ospita il cluster Kubernetes (ad esempio, IBM Cloud).
2. Raccogliere le informazioni sulla versione di Kubernetes
   Ad esempio, eseguire `kubectl version -o yaml > kubeversion.yaml`
3. Raccogliere risorse YAML che descrivono gli elementi chiave di come IBM MQ viene distribuito.

   Includere quanto segue:

   a. StatefulSet

   b. Pod (per ogni istanza di gestore di code)

   c. Servizio

   d. PersistentVolumeClaim

   e. ConfigMaps

   Ad esempio, eseguire `kubectl get pod` *PodName* `-o yaml` per ogni Pod del gestore di code.
4. Raccogliere una descrizione di ogni Pod, compresi gli eventi.
   Ad esempio, eseguire `kubectl describe` pod per ogni pod di gestione delle code.
5. Raccogliere tutti i log del Pod
   Ad esempio, eseguire `kubectl logs` *PodName* per ogni Pod del gestore di code.
6. Raccogliere l'output del comando **runmqras**

Usare `kubectl exec` per eseguire il comando **`runmqras`** all'interno di ogni contenitore del gestore di code.

Attività correlate

Raccolta di informazioni per la risoluzione dei problemi

OpenShift CP4I Informazioni di riferimento per IBM MQ nei contenitori

IBM MQ fornisce un operatore Kubernetes , che fornisce un'integrazione nativa con Red Hat OpenShift Container Platform.

OpenShift CP4I Licenze e riferimenti alle API per IBM MQ Operator

IBM MQ fornisce un Kubernetes Operatore, che offre un'integrazione nativa con Red Hat OpenShift Container Platform. L'API `v1beta1` può essere utilizzata per creare e gestire le risorse del gestore di code.

mq.ibm.com/v1beta1: Versioni di licenza attuali

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-CYPF-CRPF3H`	`Production` o `NonProduction`	IBM Cloud Pak for Integration 16.1.2	9.4.3
`L-MQQP-KBWMYE`	`Production` o `NonProduction`	IBM Cloud Pak for Integration Edizione limitata 16.1.2	9.4.3
`L-NUUP-23NH8Y`	`Production`	IBM MQ Advanced 9.4 - 02/2025	9.4.3 o 9.4.2
`L-QWBN-T3PKLE`	`NonProduction`	IBM MQ Advanced for Non-Production Environment 9.4 - 02/2025	9.4.3 o 9.4.2
`L-HYGL-6STWD6`	`Development`	IBM MQ Advanced for Developers (Non coperto da garanzia) 9.4 - 10/2024	9.4.3, 9.4.2, o 9.4.1
`L-QYVA-B365MB`	`Production` o `NonProduction`	IBM Cloud Pak for Integration 16.1.1	9.4.2 oppure 9.4.1
`L-JVML-UFQVM4`	`Production` o `NonProduction`	IBM Cloud Pak for Integration Edizione limitata 16.1.1	9.4.2 oppure 9.4.1
`L-JTPV-KYG8TF`	`Production` o `NonProduction`	IBM Cloud Pak for Integration 16.1.0	9.9.4.1 o 9.4.0
`L-BMSF-5YDSLR`	`Production` o `NonProduction`	IBM Cloud Pak for Integration Edizione limitata 16.1.0	9.9.4.1 o 9.4.0
`L-EHXT-MQCRN9`	`Production`	IBM MQ Advanced 9.4	9.9.4.1 o 9.4.0
`L-DRNQ-NGCA76`	`NonProduction`	IBM MQ Advanced for Non-Production Environment 9.4	9.4.1
`L-CLXQ-ADXTK3`	`Development`	IBM MQ Advanced for Developers (non giustificato) 9.4	9.9.4.1 o 9.4.0

Si noti che viene specificata la licenza *versione*, che non è sempre la stessa della versione di IBM MQ.

mq.ibm.com/v1beta1: Versioni di licenza precedenti

Vedere Versioni di licenza precedenti nella documentazione IBM MQ 9.3.

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. IBM MQ Documentazione: https://ibm.biz/BdPZqj. Riferimento della licenza: https://ibm.biz/Bdm9be.

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` QueueManagerStatus	Lo stato osservato di QueueManager.

.spec

Lo stato desiderato del QueueManager.

Viene visualizzato in:

- "QueueManager" a pagina 249

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.
`imagePullSecrets`LocalObjectReference array	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, vedere https://kubernetes.io/docs/concepts/containers/images#specifying-imagepullsecrets-on-a-pod.

Campo	Descrizione
`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` QueueManagerConfig	Impostazioni per il contenitore Gestore code e il gestore code sottostante.
`securityContext` Contesto di sicurezza	Impostazioni di protezione da aggiungere al securityContextdi Queue Manager Pod.
`service` Servizio	Impostazioni che si applicano a tutti i servizi di QueueManager. La modifica o la rimozione di questa sezione o dei campi al suo interno può comportare la rigenerazione di tutti i servizi di QueueManager. La rigenerazione del servizio interromperà le comunicazioni con conseguenti tempi di inattività per tutti i tipi di QueueManager, compreso NativeHA. Ai servizi rigenerati vengono assegnati nuovi indirizzi ClusterIP da Kubernetes. Richiede il sito MQ Operator 3.6.0 o superiore.
`telemetry` Telemetria	Impostazioni per la configurazione di Open Telemetry. Richiede l'operatore MQ 2.2.0 o superiore.
`template` Modello	Templating avanzato per risorse Kubernetes . Il modello consente agli utenti di sovrascrivere il modo in cui IBM MQ genera le risorse Kubernetes sottostanti, come StatefulSet, Pod e Servizi. 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.
`terminationGracePeriod Seconds` 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 .
`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` WebServerConfig	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 249

.spec.imagePullSecrets

LocalObjectReference contiene informazioni sufficienti per individuare l'oggetto referenziato all'interno dello stesso spazio dei nomi.

Viene visualizzato in:

- “.spec” a pagina 249

Campo	Descrizione
`name` Stringa	Nome del referente. Questo campo è effettivamente obbligatorio, ma a causa della retrocompatibilità può essere vuoto. Le istanze di questo tipo con un valore vuoto sono quasi certamente errate. Per saperne di più: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names. Predefinito: " ".

.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 249

.spec.license

Impostazioni che controllano l'accettazione della licenza e quali metriche di licenza utilizzare.

Viene visualizzato in:

- “.spec” a pagina 249

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. Vedere https://ibm.biz/Bdm9be per i valori validi.
`metric` Stringa	Impostazione che specifica quale metrica di licenza utilizzare. Ad esempio, `ProcessorValueUnit`, `VirtualProcessorCore`, `ManagedVirtualServer` o FREE. L'impostazione predefinita è `ProcessorValueUnit` quando si usa una licenza MQ Advanced o una licenza MQ Advanced for Non-Production, `VirtualProcessorCore` quando si usa una licenza Cloud Pak for Integration e FREE quando si usa una licenza MQ Advanced for Developers.
`use` Stringa	Impostazione che controlla il modo in cui verrà utilizzato il software, dove la licenza supporta più utilizzi. Vedere https://ibm.biz/Bdm9be 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 249

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 251

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 252

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 251

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 252

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 249

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.
`env` Matrice Env	Impostazioni per la fornitura di variabili di ambiente per il Queue Manager. Richiede l' MQ. Operatore 3.5.0 e o superiore.
`files` Matrice di file	Impostazioni per la fornitura di file per il Queue Manager. Richiede l' MQ. Operatore 3.5.0 e o superiore.
`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` QueueManagerMetrics	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 del gestore code che non sia conforme alle regole Kubernetes per i nomi (ad esempio, un nome che include lettere maiuscole).
`readinessProbe` QueueManagerReadinessProbe	Impostazioni che controllano il probe di disponibilità.
`recoveryLogs` Registri di recupero	Impostazioni per i log di ripristino di MQ . Richiede l'operatore MQ 2.4.0 o superiore.
`resources` Risorse	Impostazioni che controllano i requisiti delle risorse.

Campo	Descrizione
`route` Instrada	Impostazioni per l'instradamento del gestore code. Richiede l'operatore MQ 1.4.0 o superiore.
`startupProbe` Prova di avviamento	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.disponibilità

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 253

Campo	Descrizione
`nativeHAGroups` Gruppi nativi	Impostazioni per i gruppi HA nativi. Richiede che `type of availability` sia `NativeHA`. Richiede l' MQ. Operatore 3.5.0 e o superiore.
`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`. Vedere https://ibm.biz/Bdm9TW per maggiori dettagli.
`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.nativeHAGroups

Impostazioni per i gruppi HA nativi. Richiede che `type of availability` sia `NativeHA`. Richiede l' MQ. Operatore 3.5.0 e o superiore.

Viene visualizzato in:

- ".spec.queueManager.disponibilità" a pagina 254

Campo	Descrizione
`local` Locale	Impostazioni per questo gruppo di NativeHA.
`remotes` Serie di telecomandi	Impostazioni per la connessione a gruppi di tel NativeHA.

.spec.queueManager.availability.nativeHAGroups.locale

Impostazioni per questo gruppo di NativeHA.

Viene visualizzato in:

- “.spec.queueManager.availability.nativeHAGroups” a pagina 254

Campo	Descrizione
`name` Stringa	Il nome di questo gruppo di NativeHA.
`role` Stringa	NativeHA Il ruolo che questo gruppo di lavoro dovrebbe assumere. Live o Recovery. I valori validi sono: `"Live"; "Recovery"`.
`route` Instrada	Impostazioni per il percorso del gruppo di lavoro (NativeHA).
`tls` TL	Impostazioni TLS per configurare una comunicazione sicura tra gruppi di utenti (NativeHA). Richiesto se TLS non è configurato nella sezione disponibilità principale.

.spec.queueManager.availability.nativeHAGroups.local.route

Impostazioni per il percorso del gruppo di lavoro (NativeHA).

Viene visualizzato in:

- “.spec.queueManager.availability.nativeHAGroups.locale” a pagina 254

Campo	Descrizione
`enabled` booleano	Indica se abilitare o meno l'instradamento. L'impostazione predefinita è true.

.spec.queueManager.availability.nativeHAGroups.local.tls

Impostazioni TLS per configurare una comunicazione sicura tra gruppi di utenti (NativeHA). Richiesto se TLS non è configurato nella sezione disponibilità principale.

Viene visualizzato in:

- “.spec.queueManager.availability.nativeHAGroups.locale” a pagina 254

Campo	Descrizione
`cipherSpec` Stringa	Il nome dell' CipherSpec e per il gruppo TLS (NativeHA).
`key` Chiave	Impostazioni per la chiave TLS utilizzata nella comunicazione sicura tra gruppi Native HA.

.spec.queueManager.availability.nativeHAGroups.local.tls.key

Impostazioni per la chiave TLS utilizzata nella comunicazione sicura tra gruppi Native HA.

Viene visualizzato in:

- “.spec.queueManager.availability.nativeHAGroups.local.tls” a pagina 255

Campo	Descrizione
`secretName` Stringa	Il nome del segreto Kubernetes . Il segreto deve essere di tipo ' kubernetes.io/tls ' contenente una singola chiave privata, un certificato pubblico e una catena di fiducia.

.spec.queueManager.availability.nativeHAGroups.telecomandi

Viene visualizzato in:

- “.spec.queueManager.availability.nativeHAGroups” a pagina 254

Campo	Descrizione
`addresses` schiera	Una serie di nomi host e indirizzi di porta per comunicare con il gruppo di NativeHA e remoto. Efficace solo quando il ruolo del gruppo locale è " `Live`".
`enabled` booleano	Se iniziare o meno la comunicazione con il telecomando. Compatibile solo con il gruppo di NativeHA locale con ruolo "Live". Se non impostato, il contenitore " MQ " utilizzerà il valore predefinito interno.
`name` Stringa	Il nome del gruppo di tel NativeHA. Deve corrispondere al nome del gruppo di cluster remoto (NativeHA) configurato nel cluster remoto.
`retries` Riprova	Configurazione per il comportamento di riprova per il telecomando.
`statusInterval` intero	L'intervallo di tempo più lungo in millisecondi che il gruppo di recupero attenderà prima di inviare uno stato del registro di recupero.
`trust` Configurazione Trust	Certificati da aggiungere al repository delle chiavi del Queue Manager per la comunicazione con il gruppo di gestione remota dell' NativeHA.

.spec.queueManager.availability.nativeHAGroups.remotes.retries

Configurazione per il comportamento di riprova per il telecomando.

Viene visualizzato in:

- ".spec.queueManager.availability.nativeHAGroups.telecomandi" a pagina 255

Campo	Descrizione
`long` Lunga	Riprova configurazione per la fase di riprova lunga.
`reporting` Stringa	Se segnalare tutti i tentativi di riprova o solo la fase di riprova.
`short` Breve	Riprova configurazione per la fase di riprova breve.

.spec.queueManager.availability.nativeHAGroups.remotes.retries.long

Riprova configurazione per la fase di riprova lunga.

Viene visualizzato in:

- ".spec.queueManager.availability.nativeHAGroups.remotes.retries" a pagina 256

Campo	Descrizione
`interval` intero	Intervallo in millisecondi da attendere tra i tentativi di connessione al telecomando durante la lunga fase di ripetizione.

.spec.queueManager.availability.nativeHAGroups.remotes.retries.short

Riprova configurazione per la fase di riprova breve.

Viene visualizzato in:

- ".spec.queueManager.availability.nativeHAGroups.remotes.retries" a pagina 256

Campo	Descrizione
`count` intero	Il numero di tentativi di connessione durante la breve fase di riprova.
`interval` intero	Intervallo in millisecondi da attendere tra i tentativi di connessione al telecomando durante la breve fase di riprova.

.spec.queueManager.availability.nativeHAGroups.remotes.trust

NativeHAGroupTrust definisce una fonte di chiavi pubbliche dell'autorità di certificazione.

Viene visualizzato in:

- ".spec.queueManager.availability.nativeHAGroups.telecomandi" a pagina 255

Campo	Descrizione
`configMap` Configmap	Fornire un certificato CA utilizzando una mappa di configurazione di Kubenetes.
`secret` Segreto	Fornire un certificato CA utilizzando un segreto di certificazione (Kubernetes).

.spec.queueManager.availability.nativeHAGroups.remotes.trust.configMap

Fornire un certificato CA utilizzando una mappa di configurazione di Kubenetes.

Viene visualizzato in:

- ".spec.queueManager.availability.nativeHAGroups.remotes.trust" a pagina 257

Campo	Descrizione
`items` schiera	Chiavi all'interno dell' Kubernetes a configmap sorgente che dovrebbe essere applicata. Se non specificato, utilizza tutti i tasti.
`name` Stringa	Il nome dell'origine configmap dell' Kubernetes.

.spec.queueManager.availability.nativeHAGroups.remotes.trust.secret

Fornire un certificato CA utilizzando un segreto di certificazione (Kubernetes).

Viene visualizzato in:

- ".spec.queueManager.availability.nativeHAGroups.remotes.trust" a pagina 257

Campo	Descrizione
`items` schiera	Chiavi all'interno dell' Kubernetes e segreto che dovrebbe essere applicato. Se non specificato, utilizza tutti i tasti.
`secretName` Stringa	Il nome del segreto Kubernetes .

.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.disponibilità" a pagina 254

Campo	Descrizione
`cipherSpec` Stringa	Il nome di CipherSpec per TLS NativeHA .
`secretName` Stringa	Il nome del segreto Kubernetes .

.spec.queueManager.env

EnvVar rappresenta una variabile di ambiente presente in un contenitore.

Viene visualizzato in:

- “.spec.queueManager” a pagina 253

Campo	Descrizione
name Stringa	Nome della variabile di ambiente. Deve essere un C_IDENTIFIER.
value Stringa	I riferimenti variabili $(VAR_NAME) vengono espansi utilizzando le variabili di ambiente definite in precedenza nel contenitore e le variabili di ambiente di servizio. Se una variabile non può essere risolta, il riferimento nella stringa di input rimarrà invariato. I doppi $$ vengono ridotti a un singolo $, il che consente di evitare la sintassi $(VAR_NAME): ad esempio $$(VAR_NAME) produrrà il letterale di stringa $(VAR_NAME). I riferimenti mancanti non verranno mai espansi, indipendentemente dal fatto che la variabile esista o meno. Impostazione predefinita: .
valueFrom Valore da	Origine del valore della variabile di ambiente. Non può essere utilizzato se il valore non è vuoto.

.spec.queueManager.env.valueFrom

Origine del valore della variabile di ambiente. Non può essere utilizzato se il valore non è vuoto.

Viene visualizzato in:

- “.spec.queueManager.env” a pagina 257

Campo	Descrizione
configMapKeyRef Configmapkeyref	Seleziona un tasto di un ConfigMap.
fieldRef Campo di riferimento	Seleziona un campo del pod: supporta metadata.name, metadata.namespace, metadata.labels['<KEY>'], metadata.annotations['<KEY>'], spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.
resourceFieldRef Riferimento campo risorsa	Seleziona una risorsa del contenitore: attualmente sono supportati solo i limiti e le richieste di risorse (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory e requests.ephemeral-storage).
secretKeyRef Segretezza della chiave di lettura	Seleziona una chiave di un segreto nello spazio dei nomi del pod.

.spec.queueManager.env.valueFrom.configMapKeyRef

Seleziona un tasto di un ConfigMap.

Viene visualizzato in:

- “.spec.queueManager.env.valueFrom” a pagina 258

Campo	Descrizione
key Stringa	La chiave per selezionare.
name Stringa	Nome del referente. Questo campo è effettivamente obbligatorio, ma a causa della retrocompatibilità può essere vuoto. Le istanze di questo tipo con un valore vuoto sono quasi certamente errate. Per saperne di più: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names. Predefinito: " ".
optional booleano	Specificare se deve essere definito l' ConfigMap e o la sua chiave.

.spec.queueManager.env.valueFrom.fieldRef

Seleziona un campo del pod: supporta metadata.name, metadata.namespace, `metadata.labels['<KEY>']`, `metadata.annotations['<KEY>']`, spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

Viene visualizzato in:

- ".spec.queueManager.env.valueFrom" a pagina 258

Campo	Descrizione
`apiVersion` Stringa	La versione dello schema in cui è scritto l' FieldPath, è predefinita su `v1`.
`fieldPath` Stringa	Percorso del campo da selezionare nella versione API specificata.

.spec.queueManager.env.valueFrom.resourceFieldRef

Seleziona una risorsa del contenitore: attualmente sono supportati solo i limiti e le richieste di risorse (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory e requests.ephemeral-storage).

Viene visualizzato in:

- ".spec.queueManager.env.valueFrom" a pagina 258

Campo	Descrizione
`containerName` Stringa	Nome contenitore: obbligatorio per i volumi, facoltativo per le variabili ambientali.
`divisor`	Specifica il formato di output delle risorse esposte, il valore predefinito è `1`.
`resource` Stringa	Richiesto: risorsa da selezionare.

.spec.queueManager.env.valueFrom.secretKeyRef

Seleziona una chiave di un segreto nello spazio dei nomi del pod.

Viene visualizzato in:

- ".spec.queueManager.env.valueFrom" a pagina 258

Campo	Descrizione
`key` Stringa	La chiave del segreto per scegliere. Deve essere una chiave segreta valida.
`name` Stringa	Nome del referente. Questo campo è effettivamente obbligatorio, ma a causa della retrocompatibilità può essere vuoto. Le istanze di questo tipo con un valore vuoto sono quasi certamente errate. Per saperne di più: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names. Predefinito: `""`.
`optional` booleano	Specificare se deve essere definito il segreto o la relativa chiave.

.spec.queueManager.file

Fonte dei file.

Viene visualizzato in:

- ".spec.queueManager" a pagina 253

Campo	Descrizione
`configMap` Configmap	ConfigMap ConfigMap rappresenta un file system (Kubernetes) che contiene file.
`secret` Segreto	Kubernetes e rappresenta un e che contiene file.

.spec.queueManager.files.configMap

ConfigMap ConfigMap rappresenta un file system (Kubernetes) che contiene file.

Viene visualizzato in:

- ".spec.queueManager.file" a pagina 259

Campo	Descrizione
`defaultMountPath` Stringa	Il percorso di montaggio predefinito da utilizzare per le chiavi nell'origine dell' Kubernetes. Questo può essere ignorato specificando percorsi di montaggio individuali per le chiavi. Se non specificato, verrà utilizzato `/etc/mqm/customfiles` .
`items` Articoli in assortimento	I tasti all'interno dell' Kubernetes no la fonte che dovrebbe essere applicata.
`name` Stringa	Il nome dell'origine Kubernetes .

.spec.queueManager.files.configMap.articoli

Viene visualizzato in:

- ".spec.queueManager.files.configMap" a pagina 260

Campo	Descrizione
`item` Stringa	Il nome della chiave.
`mountPath` Stringa	Il percorso di montaggio da utilizzare per la chiave, sostituisce il percorso di montaggio predefinito.

.spec.queueManager.files.secret

Kubernetes e rappresenta un e che contiene file.

Viene visualizzato in:

- ".spec.queueManager.file" a pagina 259

Campo	Descrizione
`defaultMountPath` Stringa	Il percorso di montaggio predefinito da utilizzare per le chiavi nell'origine dell' Kubernetes. Questo può essere ignorato specificando percorsi di montaggio individuali per le chiavi. Se non specificato, verrà utilizzato `/etc/mqm/customfiles` .
`items` Articoli in assortimento	I tasti all'interno dell' Kubernetes no la fonte che dovrebbe essere applicata.
`name` Stringa	Il nome dell'origine Kubernetes .

.spec.queueManager.files.secret.items

Viene visualizzato in:

- ".spec.queueManager.files.secret" a pagina 260

Campo	Descrizione
`item` Stringa	Il nome della chiave.
`mountPath` Stringa	Il percorso di montaggio da utilizzare per la chiave, sostituisce il percorso di montaggio predefinito.

.spec.queueManager.ini

Origine dei file di configurazione INI.

Viene visualizzato in:

- ".spec.queueManager" a pagina 253

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 261

Campo	Descrizione
`items` schiera	I tasti all'interno dell' Kubernetes no la fonte che dovrebbe essere applicata.
`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 261

Campo	Descrizione
`items` schiera	I tasti all'interno dell' Kubernetes no la fonte che dovrebbe essere applicata.
`name` Stringa	Il nome dell'origine Kubernetes .

.spec.queueManager.livenessProbe

Impostazioni che controllano il probe di attività.

Viene visualizzato in:

- ".spec.queueManager" a pagina 253

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 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.
`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.metrica

Impostazioni per le metriche in stile Prometheus.

Viene visualizzato in:

- ".spec.queueManager" a pagina 253

Campo	Descrizione
`enabled` booleano	Indica se abilitare o meno un endpoint per le metriche compatibili con Prometheus. L'impostazione predefinita è true.
`serviceMonitor` Servicemonitor	Impostazioni del servizio di monitoraggio per le metriche compatibili con Prometheus. Richiede il sito MQ Operator 3.6.0 o superiore.
`tls` TL	Impostazioni TLS per configurare una comunicazione sicura per le metriche compatibili con l' Prometheus. Richiede l' MQ. Operatore 3.5.0 e o superiore.

.spec.queueManager.metrics.serviceMonitor

Impostazioni del servizio di monitoraggio per le metriche compatibili con Prometheus. Richiede il sito MQ Operator 3.6.0 o superiore.

Viene visualizzato in:

- ".spec.queueManager.metrica" a pagina 262

Campo	Descrizione
`enabled` booleano	Abilitare o meno il monitoraggio del servizio. L'impostazione predefinita è true.

.spec.queueManager.metrics.tls

Impostazioni TLS per configurare una comunicazione sicura per le metriche compatibili con l' Prometheus. Richiede l' MQ. Operatore 3.5.0 e o superiore.

Viene visualizzato in:

- ".spec.queueManager.metrica" a pagina 262

Campo	Descrizione
`provider` Stringa	Il provider da utilizzare per configurare la comunicazione sicura per le metriche compatibili con l' Prometheus. Il valore predefinito è none. Utilizzare none per metriche di sicurezza dell' HTTP. Utilizza `openshift` per utilizzare i certificati OpenShift-provided con un ServiceMonitor.

.spec.queueManager.mqsc

Origine dei file di configurazione MQSC.

Viene visualizzato in:

- ".spec.queueManager" a pagina 253

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 263

Campo	Descrizione
`items` schiera	I tasti all'interno dell' Kubernetes no la fonte che dovrebbe essere applicata.
`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 263

Campo	Descrizione
`items` schiera	I tasti all'interno dell' Kubernetes no la fonte che dovrebbe essere applicata.
`name` Stringa	Il nome dell'origine Kubernetes .

.spec.queueManager.readinessProbe

Impostazioni che controllano il probe di disponibilità.

Viene visualizzato in:

- ".spec.queueManager" a pagina 253

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 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.recoveryLogs

Impostazioni per i log di ripristino di MQ . Richiede l'operatore MQ 2.4.0 o superiore.

Viene visualizzato in:

- ".spec.queueManager" a pagina 253

Campo	Descrizione
`logFilePages` intero	I dati del log di ripristino sono contenuti in una serie di file. La dimensione del file di log è specificata in unità di pagine da 4 KB.

.spec.queueManager.risorse

Impostazioni che controllano i requisiti delle risorse.

Viene visualizzato in:

- ".spec.queueManager" a pagina 253

Campo	Descrizione
`limits` Limiti	Impostazioni CPU & memoria.
`requests` Richieste	Impostazioni CPU & memoria.

.spec.queueManager.resources.limits

Impostazioni CPU & memoria.

Viene visualizzato in:

- ".spec.queueManager.risorse" a pagina 264

Campo	Descrizione
`cpu`	
`memory`	

.spec.queueManager.resources.requests

Impostazioni CPU & memoria.

Viene visualizzato in:

- “.spec.queueManager.risorse” a pagina 264

Campo	Descrizione
cpu	
memory	

.spec.queueManager.percorso

Impostazioni per l'instradamento del gestore code. Richiede l'operatore MQ 1.4.0 o superiore.

Viene visualizzato in:

- “.spec.queueManager” a pagina 253

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 253

Campo	Descrizione
failureThreshold intero	Numero minimo di errori consecutivi per il probe da considerare non riuscito. Il valore predefinito è 24.
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.stoccaggio

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 253

Campo	Descrizione
allowVolumeExpansion booleano	Indica se consentire o meno l'espansione dei volumi.

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 archiviazione può essere di tipo ReadWriteOnce o ReadWriteMany. Se `type of availability` è `MultiInstance`, la classe di archiviazione 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 un proprio valore per deleteClaim che sovrascrive l'impostazione defaultDeleteClaim. 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.
`scratch` Svuota	Impostazioni per il volume temporaneo vuoto del gestore code. Questo volume verrà montato come cartella '/run' sul contenitore. Applicabile solo se il file system root è impostato su sola lettura. Richiede l'operatore MQ 3.0.0 o superiore.
`tmp` Tmp	Impostazioni per il volume temporaneo Tmp del gestore code. Questo volume verrà montato sul contenitore come cartella '/tmp'. I file di dati diagnostici, come il file zip prodotto dal comando runmqras, verranno creati in questo volume. Applicabile solo se il file system root è impostato su sola lettura. Richiede l'operatore MQ 3.0.0 o superiore.

.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.stoccaggio" a pagina 265

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 archiviazione può essere di tipo ReadWriteOnce o ReadWriteMany. Se `type of availability` è `MultiInstance`, la classe di archiviazione 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` e il filesystem root è impostato su sola lettura. Richiede l'operatore MQ 3.0.0 o superiore.
`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.stoccaggio" a pagina 265

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 archiviazione può essere di tipo ReadWriteOnce o ReadWriteMany. Se `type of availability` è `MultiInstance`, la classe di archiviazione 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` e il filesystem root è impostato su sola lettura. Richiede l'operatore MQ 3.0.0 o superiore.
`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.stoccaggio" a pagina 265

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 archiviazione può essere di tipo ReadWriteOnce o ReadWriteMany. Se `type of availability` è `MultiInstance`, la classe di archiviazione 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` e il filesystem root è impostato su sola lettura. Richiede l'operatore MQ 3.0.0 o superiore.
`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.scratch

Impostazioni per il volume temporaneo vuoto del gestore code. Questo volume verrà montato come cartella '/run' sul contenitore. Applicabile solo se il file system root è impostato su sola lettura. Richiede l'operatore MQ 3.0.0 o superiore.

Viene visualizzato in:

- ".spec.queueManager.stoccaggio" a pagina 265

Campo	Descrizione
`sizeLimit` Stringa	Limite di dimensione del volume effimero, incluse le unità SI. Ad esempio, `2Gi`. Valido solo quando il filesystem root è impostato su sola lettura. Richiede l'operatore MQ 3.0.0 o superiore.

.spec.queueManager.storage.tmp

Impostazioni per il volume temporaneo Tmp del gestore code. Questo volume verrà montato sul contenitore come cartella '/tmp'. I file di dati diagnostici, come il file zip prodotto dal comando runmqras, verranno creati in questo volume. Applicabile solo se il file system root è impostato su sola lettura. Richiede l'operatore MQ 3.0.0 o superiore.

Viene visualizzato in:

- ".spec.queueManager.stoccaggio" a pagina 265

Campo	Descrizione
`sizeLimit` Stringa	Limite di dimensione del volume effimero, incluse le unità SI. Ad esempio, `2Gi`. Valido solo quando il filesystem root è impostato su sola lettura. Richiede l'operatore MQ 3.0.0 o superiore.

.spec.securityContext

Impostazioni di protezione da aggiungere al securityContextdi Queue Manager Pod.

Viene visualizzato in:

- ".spec" a pagina 249

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 con 0660. Se non impostato, il Kubelet non modificherà la proprietà e i permessi di nessun 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.
`readOnlyRootFilesystem` booleano	Indica se abilitare o meno le impostazioni del file system root di sola lettura per il gestore code. Il valore predefinito è false. Richiede l'operatore MQ 3.0.0 o superiore.
`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.service

Impostazioni che si applicano a tutti i servizi di QueueManager. La modifica o la rimozione di questa sezione o dei campi al suo interno può comportare la rigenerazione di tutti i servizi di QueueManager. La rigenerazione del servizio interromperà le comunicazioni con conseguenti tempi di inattività per tutti i tipi di QueueManager, compreso NativeHA. Ai servizi rigenerati vengono assegnati nuovi indirizzi ClusterIP da Kubernetes. Richiede il sito MQ Operator 3.6.0 o superiore.

Viene visualizzato in:

- ".spec" a pagina 249

Campo	Descrizione
`ipFamilies` schiera	Specifica le famiglie IP da utilizzare per tutti i servizi QueueManager. Le opzioni valide sono IPv4 e IPv6. Se non specificato, Kubernetes imposterà un valore predefinito basato sull'impostazione di ipFamilyPolicy. Se impostato manualmente, la creazione del servizio fallirà se la famiglia IP non è disponibile nel cluster o se i valori sono incompatibili con l'impostazione di ipFamilyPolicy. Le modifiche al primo valore dell'array (la famiglia IP primaria) o la rimozione di questo campo comporteranno la rigenerazione di tutti i servizi QueueManager. La rigenerazione del servizio interromperà le comunicazioni con conseguenti tempi di inattività per tutti i tipi di QueueManager, compreso NativeHA. Ai servizi rigenerati vengono assegnati nuovi indirizzi ClusterIP da Kubernetes.

Campo	Descrizione
`ipFamilyPolicy` Stringa	Specifica il criterio della famiglia IP da utilizzare per tutti i servizi QueueManager. Le opzioni valide sono SingleStack, PreferDualStack e RequireDualStack. SingleStack è per una singola famiglia IP. PreferDualStack è per due famiglie IP su cluster configurati dual-stack o per una singola famiglia IP su cluster single-stack. RequireDualStack è per due famiglie IP su cluster configurati in dual-stack, altrimenti fallisce. La rimozione di questo campo comporta la rigenerazione di tutti i servizi di QueueManager. La rigenerazione del servizio interromperà le comunicazioni con conseguenti tempi di inattività per tutti i tipi di QueueManager, compreso NativeHA. Ai servizi rigenerati vengono assegnati nuovi indirizzi ClusterIP da Kubernetes.

.spec.telemetry

Impostazioni per la configurazione di Open Telemetry. Richiede l'operatore MQ 2.2.0 o superiore.

Viene visualizzato in:

- ".spec" a pagina 249

Campo	Descrizione
`tracing` Traccia	Impostazioni per la traccia Open Telemetry.

.spec.telemetry.tracing

Impostazioni per la traccia Open Telemetry.

Viene visualizzato in:

- ".spec.telemetry" a pagina 270

Campo	Descrizione
`instana` Instana	Impostazioni per la traccia Instana.

.spec.telemetry.tracing.instana

Impostazioni per la traccia Instana.

Viene visualizzato in:

- ".spec.telemetry.tracing" a pagina 270

Campo	Descrizione
`agentHost` Stringa	Il nome host dell'agent Instana a cui inviare i dati di traccia. Questo non deve includere un protocollo.
`enabled` booleano	Indica se abilitare o meno la traccia Instana. Il valore predefinito è false.
`protocol` Stringa	Il protocollo da utilizzare nella comunicazione con l'agente Instana. http e https sono supportati.

.spec.template

Templating avanzato per risorse Kubernetes . Il modello consente agli utenti di sovrascrivere il modo in cui IBM MQ genera le risorse Kubernetes sottostanti, come StatefulSet, Pod e Servizi. 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 249

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 249

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 271

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 271

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 271

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 271

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 272

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 272

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 249

Campo	Descrizione
`console` Console	Impostazioni per la console Web di MQ . Richiede l'operatore MQ 3.0.0 o superiore.
`enabled` booleano	Indica se abilitare o meno il server Web. Il valore predefinito è false.
`manualConfig` Configurazione manuale	Impostazioni per fornire la configurazione XML del server Web. Richiede l'operatore MQ 3.0.0 o superiore.
`route` Instrada	Impostazioni per il percorso del server web. Richiede il sito MQ Operator 3.6.0 o superiore.

.spec.web.console

Impostazioni per la console Web di MQ . Richiede l'operatore MQ 3.0.0 o superiore.

Viene visualizzato in:

- ".spec.web" a pagina 274

Campo	Descrizione
`authentication` Autenticazione	Impostazioni di autenticazione per la console Web di MQ . Richiede l'operatore MQ 3.0.0 o superiore.
`authorization` Autorizzazione	Impostazioni di autorizzazione per la console Web di MQ . Richiede l'operatore MQ 3.0.0 o superiore.

.spec.web.console.authentication

Impostazioni di autenticazione per la console Web di MQ . Richiede l'operatore MQ 3.0.0 o superiore.

Viene visualizzato in:

- ".spec.web.console" a pagina 274

Campo	Descrizione
`provider` Stringa	Il fornitore di autenticazione da utilizzare per la console Web di MQ . Utilizza `integration-keycloak` per utilizzare SSO (single sign - on) con la IU della piattaforma Cloud Pak for Integration (Keycloak). Il valore predefinito è `integration-keycloak` se si utilizza una licenza Cloud Pak for Integration o `manual` se si utilizza una licenza MQ . Utilizzare `manual` se si desidera fornire la propria configurazione.

.spec.web.console.authorization

Impostazioni di autorizzazione per la console Web di MQ . Richiede l'operatore MQ 3.0.0 o superiore.

Viene visualizzato in:

- ".spec.web.console" a pagina 274

Campo	Descrizione
`provider` Stringa	Il provider di autorizzazione da utilizzare per la console Web di MQ . Utilizza `integration-keycloak` per utilizzare i ruoli forniti da Cloud Pak for Integration Keycloak. Utilizzare `manual` se si desidera fornire la propria configurazione. Il valore predefinito è `integration-keycloak` se si utilizza una licenza Cloud Pak for Integration o `manual` se si utilizza una licenza MQ .

.spec.web.manualConfig

Impostazioni per fornire la configurazione XML del server Web. Richiede l'operatore MQ 3.0.0 o superiore.

Viene visualizzato in:

- ".spec.web" a pagina 274

Campo	Descrizione
`configMap` Configmap	ConfigMap rappresenta una ConfigMap Kubernetes che contiene la configurazione XML del server Web.
`secret` Segreto	Il segreto rappresenta un segreto Kubernetes che contiene la configurazione XML del server web. L'utilizzo di un segreto protegge le credenziali nel livello Kubernetes , ma è possibile che gli strumenti di monitoraggio o di risoluzione dei problemi possano esporre il file sottostante in modo non sicuro. Per una maggiore sicurezza, codificare le credenziali utilizzando "securityUtility"

.spec.web.manualConfig.configMap

ConfigMap rappresenta una ConfigMap Kubernetes che contiene la configurazione XML del server Web.

Viene visualizzato in:

- ".spec.web.manualConfig" a pagina 275

Campo	Descrizione
`name` Stringa	Il nome dell'origine Kubernetes .

.spec.web.manualConfig.segreto

Il segreto rappresenta un segreto Kubernetes che contiene la configurazione XML del server web. L'utilizzo di un segreto protegge le credenziali nel livello Kubernetes , ma è possibile che gli strumenti di monitoraggio o di risoluzione dei problemi possano esporre il file sottostante in modo non sicuro. Per una maggiore sicurezza, codificare le credenziali utilizzando "securityUtility"

Viene visualizzato in:

- ".spec.web.manualConfig" a pagina 275

Campo	Descrizione
`name` Stringa	Il nome dell'origine Kubernetes .

.spec.web.route

Impostazioni per il percorso del server web. Richiede il sito MQ Operator 3.6.0 o superiore.

Viene visualizzato in:

- ".spec.web" a pagina 274

Campo	Descrizione
`enabled` booleano	Indica se abilitare o meno l'instradamento. L'impostazione predefinita è true.

.stato

Lo stato osservato di QueueManager.

Viene visualizzato in:

- "QueueManager" a pagina 249

Campo	Descrizione
`adminUiUrl` Stringa	URL per l'interfaccia amministrativa.
`availability` Disponibilità	Lo stato di disponibilità per il gestore code.
`conditions`QueueManagerStatusCondition array	Le condizioni rappresentano le ultime osservazioni disponibili dello stato del gestore code.
`customImages` booleano	Se l'immagine di Queue Manager selezionata dall'operatore è stata sostituita o meno con un'immagine personalizzata.
`endpoints`QueueManagerStatusEndpoint array	Informazioni sugli endpoint che questo gestore code sta esponendo, come gli endpoint API o UI.
`metadata` Metadati	I metadati rappresentano ulteriori informazioni per il gestore code, incluso lo statoKeycloak dell'integrazione.
`name` Stringa	Il nome del gestore code.
`nativeHACRR` Nativohacrr	Informazioni sulla replica interregionale nativa HA configurata.
`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 276

Campo	Descrizione
`initialQuorumEstablished` booleano	Se è stato stabilito o meno un quorum iniziale per NativeHA.

.status.conditions

QueueManagerStatusCondition definisce le condizioni del Queue Manager.

Viene visualizzato in:

- ".stato" a pagina 276

Campo	Descrizione
`lastTransitionTime` Stringa	L'ultima volta in cui la condizione è passata da un stato all'altro.
`message` Stringa	Messaggio leggibile che indica i dettagli sull'ultima transizione.

Campo	Descrizione
`reason` Stringa	Motivo dell'ultima transizione di questo stato.
`status` Stringa	Stato della condizione. I valori validi sono: `"True"; "False"; "Unknown"`.
`type` Stringa	Tipo di condizione.

.status.endpoints

QueueManagerStatusEndpoint definisce gli endpoint del QueueManager.

Viene visualizzato in:

- ".stato" a pagina 276

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.metadata

I metadati rappresentano ulteriori informazioni per il gestore code, incluso lo statoKeycloak dell'integrazione.

Viene visualizzato in:

- ".stato" a pagina 276

Campo	Descrizione
`integrationKeycloak` Integrazione: keycloak	QueueManagerStatusIntegrationKeycloak definisce lo stato di Keycloak per il QueueManager.

.status.metadata.integrationKeycloak

QueueManagerStatusIntegrationKeycloak definisce lo stato di Keycloak per il QueueManager.

Viene visualizzato in:

- ".status.metadata" a pagina 277

Campo	Descrizione
`clientName` Stringa	

.status.nativeHACRR

Informazioni sulla replica interregionale nativa HA configurata.

Viene visualizzato in:

- ".stato" a pagina 276

Campo	Descrizione
`local` Locale	Stato del cluster locale.

.status.nativeHACRR.locale

Stato del cluster locale.

Viene visualizzato in:

- ".status.nativeHACRR" a pagina 277

Campo	Descrizione
`address` Stringa	Indirizzo del cluster locale.
`groupName` Stringa	Nome del gruppo del cluster locale.

.status.versions

Versione di MQ utilizzata e altre versioni disponibili da IBM Entitled Registry.

Viene visualizzato in:

- ".stato" a pagina 276

Campo	Descrizione
`available` QueueManagerStatusVersionAvailable	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 278

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 278

Campo	Descrizione
Array `licenses` Licenses	Le licenze applicabili per questa versione di QueueManager.
`name` Stringa	Versione `name` per questa versione di QueueManager. Questi sono valori validi per il campo `spec.version` .

.status.versions.available.versions.licenses

QueueManagerStatusLicense definisce una licenza.

Viene visualizzato in:

- “.status.versions.available.versions” a pagina 278

Campo	Descrizione
`displayName` Stringa	Nome di visualizzazione per la licenza.
`link` Stringa	Link al contenuto della licenza.
`matchesCurrentType` booleano	Se la licenza corrisponde o meno al tipo di licenza attualmente utilizzata.
`name` Stringa	Nome della licenza.

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.

Le seguenti condizioni sono state definite per una risorsa `QueueManager` :

Tabella 2. Condizioni di stato del gestore code

Componente	Tipo di condizione	Codice di errore	Messaggio di avvertenza
QueueManager[2]	Bloccato	OperatorDependency	Per installare, questa istanza richiede la configurazione di Keycloak da parte di [IBM Cloud Pak for Integration]. Questa istanza rimarrà nello stato [In sospeso] finché Keycloak non viene riportato come [KeycloakReady] nella risorsa Cp4iServicesBinding per questa QueueManager.
			Per installare, questa istanza richiede l'operatore [IBM IAM]. Questa istanza rimarrà in stato [Bloccato] fino a quando l'operatore non verrà installato da [IBM Cloud Pak foundational services].
	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
	In sospeso	Arrestato	Il gestore code MQ è stato arrestato poiché l'annotazione 'mq.ibm.com/stop' è presente ed è impostata su 'true' nella definizione QueueManager . Quando viene arrestato, il conteggio di repliche di QueueManager StatefulSet è impostato a zero, rimuovendo tutti i pod del gestore code MQ .
	Errore	Non superato	Distribuzione del gestore code MQ non riuscita
	Avviso	UnsupportedVersion	Un operando è stato installato da un operatore che non è supportato nella versione OCP < *ocp_version*>. Questo operando non è supportato.
	Avviso	CP4I-LTS Supporto	Un CP4I-LTS operando < *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	CP4I-LTS Supporto	Un CP4I-LTS operando < *mq_version*> è stato installato ma la versione OCP < *ocp_version*> non si qualifica per la durata del supporto esteso. Questo operando non è idoneo per la durata di supporto esteso.

[2] Le condizioni `Creating` e `Failed` monitorano l'avanzamento generale della distribuzione del gestore code. Se stai usando una licenza IBM Cloud Pak for Integration e la console web è abilitata, la condizione `OidcPending` registra lo stato del gestore code in attesa del completamento della registrazione del client OIDC con IAM.

Tabella 2. Condizioni di stato del gestore code (Continua)			
Componente	**Tipo di condizione**	**Codice di errore**	**Messaggio di avvertenza**
Pod[3]	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[4]	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
	Avviso	StorageExpansionin sospeso	L'espansione del volume è in sospeso per le seguenti PVC [< elenco di pvcs>]
	Avviso	StorageMismatch	Le dimensioni di memoria definite nella risorsa QueueManager non corrispondono alla capacità di una o più PVC di cui è stato eseguito il provisioning [< elenco di pvcs>]. AllowVolumeExpansion è impostato su false nella risorsa QueueManager , quindi l'operatore MQ non tenterà di riconciliare queste differenze.
	Errore	StorageFailed	Impossibile eseguire il provisioning della memoria per il gestore code MQ

Linux IBM MQ annotazioni sulla licenza del contenitore

Negli ambienti Kubernetes e Red Hat OpenShift Container Platform , IBM License Service usa le annotazioni di Pod per tracciare l'uso in base ai limiti definiti sul contenitore, piuttosto che sulla macchina sottostante. IBM MQ Operator imposta queste annotazioni di Pod per l'utente, ma quando si distribuisce con altri mezzi, è necessario impostarle da soli.

Per informazioni su come IBM MQ Operator annota i pod in base ai valori dello YAML di un gestore di code, vedere “IBM MQ che vengono applicati dal contenitore di annotazioni di licenza IBM MQ Operator” a pagina 286.

Quando si distribuisce un contenitore IBM MQ , esistono due approcci comuni alle licenze:

- Licenza dell'intera macchina che esegue il contenitore.

[3] 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`.

[4] 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. Le condizioni di storage monitorano anche l'avanzamento delle espansioni di volume e avvisano di non corrispondenze tra le dimensioni di storage definite nella definizione del gestore code e la dimensione delle PVC distribuite. Se si verifica un errore durante il provisioning della memoria, la condizione `StorageFailed` viene aggiunta all'elenco delle condizioni e la condizione generale del gestore code viene impostata su `Failed`.

- Licenziate il contenitore in base ai limiti associati (soloKubernetes e Red Hat OpenShift Container Platform)

Entrambe le opzioni sono disponibili per i clienti e ulteriori dettagli sono disponibili alla pagina IBM Container Licenses su 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.

Nota: Molte delle annotazioni contengono una o entrambe le seguenti righe:

```
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "PROCESSOR_VALUE_UNIT" | "VIRTUAL_PROCESSOR_CORE"
```

È necessario modificare queste righe prima di utilizzare l'annotazione:

- Per `productChargedContainers`, è necessario scegliere `"All"`o sostituire il nome effettivo del contenitore.
- Per `productMetric`, è necessario scegliere uno dei valori offerti.

Annotazioni da utilizzare con una titolarità di prodotto IBM MQ

Se hai una titolarità del prodotto IBM MQ , seleziona di seguito l'annotazione che corrisponde alla titolarità che hai acquistato e che vuoi utilizzare.

- “IBM MQ” a pagina 284
- “IBM MQ Avanzate” a pagina 284
- “IBM MQ per ambiente di non produzione” a pagina 284
- “IBM MQ Avanzate per ambienti non di produzione” a pagina 284
- “IBM MQ Advanced per gli sviluppatori” a pagina 285

Le annotazioni IBM MQ da utilizzare con le configurazioni IBM MQ Multi Instance High Availability sono le seguenti. Consultare anche “Selezione delle annotazioni corrette per le configurazioni HA (High Availability)” a pagina 283.

- “Istanze multiple del contenitore IBM MQ” a pagina 285
- “IBM MQ Istanza multipla contenitore avanzato” a pagina 285
- “IBM MQ Contenitore a più istanze per l'ambiente di non produzione” a pagina 285
- “IBM MQ Istanza multipla del contenitore avanzata per ambiente di non produzione” a pagina 285

Annotazioni da utilizzare con la titolarità prodotto CP4I

Se si dispone della titolarità IBM Cloud Pak for Integration (CP4I), selezionare l'annotazione riportata di seguito che corrisponde alla titolarità acquistata e che si desidera utilizzare.

- “Titolarità IBM MQ con CP4I” a pagina 285
- “IBM MQ Avanzato con titolarità CP4I” a pagina 285
- “IBM MQ per l'ambiente di non produzione con titolarità CP4I” a pagina 285
- “IBM MQ Avanzate per ambienti non di produzione con titolarità CP4I” a pagina 286

Le annotazioni CP4I da utilizzare con le configurazioni IBM MQ Multi Instance High Availability sono le seguenti. Consultare anche “Selezione delle annotazioni corrette per le configurazioni HA (High Availability)” a pagina 283.

- “Titolarità IBM MQ Container Multi Instance with CP4I” a pagina 286
- “Titolarità IBM MQ Advanced Container Multi Instance with CP4I” a pagina 286
- “IBM MQ Container Multi Instance for Non - Production Environment con titolarità CP4I” a pagina 286
- “Titolarità IBM MQ Advanced Container Multi Instance for Non - Production Environment con CP4I” a pagina 286

Selezione delle annotazioni corrette per le configurazioni HA (High Availability)

IBM MQ Istanza multipla

Quando distribuisci una coppia di gestori code in una configurazione ad alta disponibilità a più istanze IBM MQ , devi utilizzare la stessa annotazione su entrambe le istanze. È necessario selezionare una delle seguenti annotazioni, a seconda della titolarità acquistata:

- Titolarità autonoma IBM MQ o IBM MQ Advanced
  - “Istanze multiple del contenitore IBM MQ” a pagina 285
  - “IBM MQ Istanza multipla contenitore avanzato” a pagina 285
  - “IBM MQ Contenitore a più istanze per l'ambiente di non produzione” a pagina 285
  - “IBM MQ Istanza multipla del contenitore avanzata per ambiente di non produzione” a pagina 285
- IBM Cloud Pak for Integration titolarità
  - “Titolarità IBM MQ Container Multi Instance with CP4I” a pagina 286
  - “Titolarità IBM MQ Advanced Container Multi Instance with CP4I” a pagina 286
  - “IBM MQ Container Multi Instance for Non - Production Environment con titolarità CP4I” a pagina 286
  - “Titolarità IBM MQ Advanced Container Multi Instance for Non - Production Environment con CP4I” a pagina 286

Quando vengono utilizzati con la titolarità IBM Cloud Pak for Integration , i rapporti di titolarità nelle annotazioni garantiscono che venga registrato il consumo di titolarità corretto. Quando vengono utilizzate con titolarità IBM MQ o IBM MQ Advanced autonome, le annotazioni riportate nel License Service per ogni istanza devono essere associate alle parti di titolarità IBM MQ nel modo seguente:

- IBM MQ Advanced Container Istanza multipla
  - 1 x IBM MQ Advanced **e** 1 x IBM MQ Advanced High Availability Replica **o**
  - 2 x IBM MQ Advanced[5]
- IBM MQ Advanced Container Istanza multipla per l'ambiente di non produzione
  - 1 x IBM MQ Advanced **e** 1 x IBM MQ Advanced High Availability Replica **o**
  - 2 x IBM MQ Advanced per ambiente non di produzione)[5]
- Istanze multiple del contenitore IBM MQ
  - 1 x IBM MQ **e** 1 x IBM MQ High Availability Replica **o**
  - 2 x IBM MQ[5]
- IBM MQ Contenitore a più istanze per l'ambiente di non produzione
  - 1 x IBM MQ **e** 1 x IBM MQ High Availability Replica **o**
  - 2 x IBM MQ per ambiente non di produzione) [5]

IBM MQ HA nativa

[5] Questa opzione di titolarità è sub - ottimale e deve essere utilizzata solo se non è disponibile alcuna titolarità della parte di replica ad alta disponibilità pertinente.

Se si stanno distribuendo tre gestori code in un quorum HA nativo, solo l'istanza attiva utilizza la titolarità. Tutte le istanze devono avere la stessa annotazione. A seconda della titolarità acquistata, è necessario selezionare una delle seguenti opzioni:

- Titolarità autonoma IBM MQ o IBM MQ Advanced
  - "IBM MQ Avanzate" a pagina 284
  - "IBM MQ Avanzate per ambienti non di produzione" a pagina 284
- IBM Cloud Pak for Integration titolarità
  - "IBM MQ Avanzato con titolarità CP4I" a pagina 285
  - "IBM MQ Avanzate per ambienti non di produzione con titolarità CP4I" a pagina 286

Annotazioni

Il resto di questo argomento descrive in dettaglio il contenuto di ogni annotazione.

IBM MQ

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

Per utilizzare le funzioni Native HA e Cross-Region Replication sui gestori di code che utilizzano queste annotazioni di licenza, consultare l'argomento corrispondente al proprio ambiente:

- "Configurazione dell'HA nativo e della replica interregionale sui gestori di code con licenza IBM MQ utilizzando il software IBM MQ Operator" a pagina 115
- "Configurazione di Native HA e Cross-Region Replication su gestori di code con licenza IBM MQ in Kubernetes" a pagina 186

IBM MQ Avanzate

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

IBM MQ per ambiente di non produzione

```
productID: "151bec68564a4a47a14e6fa99266deff"
productName: "IBM MQ for Non-Production Environment"
productMetric: "PROCESSOR_VALUE_UNIT" | "VIRTUAL_PROCESSOR_CORE"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
```

Per utilizzare le funzioni Native HA e Cross-Region Replication sui gestori di code che utilizzano queste annotazioni di licenza, consultare l'argomento corrispondente al proprio ambiente:

- "Configurazione dell'HA nativo e della replica interregionale sui gestori di code con licenza IBM MQ utilizzando il software IBM MQ Operator" a pagina 115
- "Configurazione di Native HA e Cross-Region Replication su gestori di code con licenza IBM MQ in Kubernetes" a pagina 186

IBM MQ Avanzate per ambienti non di produzione

```
productID: "21dfe9a0f00f444f888756d835334909"
productName: "IBM MQ Advanced for Non-Production Environment"
productMetric: "PROCESSOR_VALUE_UNIT" | "VIRTUAL_PROCESSOR_CORE"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
```

IBM MQ Advanced per gli sviluppatori

```
productID: "2f886a3eefbe4ccb89b2adb97c78b9cb"
productName: "IBM MQ Advanced for Developers (Non-Warranted)"
productMetric: "FREE"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
```

Istanze multiple del contenitore IBM MQ

```
productID: "2dea73b866b648b6b4abe2a85eb76964"
productName: "IBM MQ Container Multi Instance"
productMetric: "PROCESSOR_VALUE_UNIT" | "VIRTUAL_PROCESSOR_CORE"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
```

IBM MQ Istanza multipla contenitore avanzato

```
productID: "bd35bff411bb47c2a3f3a4590f33a8ef"
productName: "IBM MQ Advanced Container Multi Instance"
productMetric: "PROCESSOR_VALUE_UNIT" | "VIRTUAL_PROCESSOR_CORE"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
```

IBM MQ Contenitore a più istanze per l'ambiente di non produzione

```
productID: "af11b093f16a4a26806013712b860b60"
productName: "IBM MQ Container Multi Instance for Non-Production Environment"
productMetric: "PROCESSOR_VALUE_UNIT" | "VIRTUAL_PROCESSOR_CORE"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
```

IBM MQ Istanza multipla del contenitore avanzata per ambiente di non produzione

```
productID: "31f844f7a96b49749130cd0708fdbb17"
productName: "IBM MQ Advanced Container Multi Instance for Non-Production Environment"
productMetric: "PROCESSOR_VALUE_UNIT" | "VIRTUAL_PROCESSOR_CORE"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
```

Titolarità IBM MQ con CP4I

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

IBM MQ Avanzato con titolarità CP4I

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

IBM MQ per l'ambiente di non produzione con titolarità CP4I

```
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
cloudpakName: "IBM Cloud Pak for Integration"
productID: "151bec68564a4a47a14e6fa99266deff"
productName: "IBM MQ for Non-Production Environment"
productMetric: "VIRTUAL_PROCESSOR_CORE"
```

```
productChargedContainers:  "All" | "NAME_OF_CONTAINER"
productCloudpakRatio: "8:1"
```

IBM MQ Avanzate per ambienti non di produzione con titolarità CP4I

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

Titolarità IBM MQ Container Multi Instance with CP4I

```
productName: "IBM MQ Container Multi Instance"
productID: "2dea73b866b648b6b4abe2a85eb76964"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productMetric: "VIRTUAL_PROCESSOR_CORE"
productCloudpakRatio: "10:3"
cloudpakName: "IBM Cloud Pak for Integration"
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
```

Titolarità IBM MQ Advanced Container Multi Instance with CP4I

```
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
cloudpakName: "IBM Cloud Pak for Integration"
productID: "bd35bff411bb47c2a3f3a4590f33a8ef"
productName: "IBM MQ Advanced Container Multi Instance"
productMetric: "VIRTUAL_PROCESSOR_CORE"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productCloudpakRatio: "5:3"
```

IBM MQ Container Multi Instance for Non - Production Environment con titolarità CP4I

```
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
cloudpakName: "IBM Cloud Pak for Integration"
productID: "af11b093f16a4a26806013712b860b60"
productName: "IBM MQ Container Multi Instance for Non-Production Environment"
productMetric: "VIRTUAL_PROCESSOR_CORE"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productCloudpakRatio: "20:3"
```

Titolarità IBM MQ Advanced Container Multi Instance for Non - Production Environment con CP4I

```
cloudpakId: "c8b82d189e7545f0892db9ef2731b90d"
cloudpakName: "IBM Cloud Pak for Integration"
productID: "31f844f7a96b49749130cd0708fdbb17"
productName: "IBM MQ Advanced Container Multi Instance for Non-Production Environment"
productMetric: "VIRTUAL_PROCESSOR_CORE"
productChargedContainers: "All" | "NAME_OF_CONTAINER"
productCloudpakRatio: "10:3"
```

Linux IBM MQ che vengono applicati dal contenitore di annotazioni di licenza IBM MQ Operator

Negli ambienti Kubernetes e Red Hat OpenShift Container Platform , IBM License Service usa le annotazioni di Pod per tracciare l'uso in base ai limiti definiti sul contenitore, piuttosto che sulla macchina sottostante. IBM MQ Operator imposta queste annotazioni di Pod per l'utente, ma quando si distribuisce con altri mezzi, è necessario impostarle da soli.

Per le annotazioni disponibili, vedere “IBM MQ annotazioni sulla licenza del contenitore” a pagina 281. La tabella seguente descrive quali annotazioni vengono applicate a un pod da IBM MQ Operator, in base ai valori presenti nello YAML di un gestore di code:

- Il valore di `spec.license.license`
- Il valore di `spec.license.use`
- Il valore di `spec.queueManager.availability.type`

Nota: Questa tabella si riferisce alle licenze dei CD più recenti, per IBM MQ 9.4.3.0. Per le licenze di altre versioni di IBM MQ , vedere “Licenze e riferimenti alle API per IBM MQ Operator” a pagina 248.

Annotazione	**Value of `spec.license.license`**	**Valore di `spec.license.use`**	**Valore di `spec.queueManager.availability.type`**
IBM MQ Advanced	L-NUUP-23NH8Y	`Production`	`SingleInstance` o `NativeHA`
IBM MQ Advanced for Non-Production Environment	L-QWBN-T3PKLE	`NonProduction`	`SingleInstance` o `NativeHA`
IBM MQ Advanced for Developers	L-HYGL-6STWD6	`Development`	`SingleInstance` o `NativeHA`
IBM MQ Advanced Container Multi Instance	L-NUUP-23NH8Y	`Production`	`MultiInstance`
IBM MQ Advanced Container Multi Instance for Non-Production Environment	L-QWBN-T3PKLE	`NonProduction`	`MultiInstance`
IBM MQ Advancedcon CP4I diritto	L-CYPF-CRPF3H	`Production`	`SingleInstance` o `NativeHA`
IBM MQ Advanced for Non-Production Environment con CP4I diritto	L-CYPF-CRPF3H	`NonProduction`	`SingleInstance` o `NativeHA`
IBM MQ Advanced Container Multi Instance con CP4I diritto	L-CYPF-CRPF3H	`Production`	`MultiInstance`
IBM MQ Advanced Container Multi Instance for Non-Production Environment con CP4I diritto	L-CYPF-CRPF3H	`NonProduction`	`MultiInstance`

V 9.4.1 Linux IBM MQ Advanced immagine contenitore

Per IBM MQ Advanced è disponibile un'immagine del contenitore precostruita. Questa immagine è disponibile nel sito IBM Container Registry.

Utilizzo

Per poter utilizzare l'immagine, è necessario accettare i termini della licenza IBM MQ Advanced impostando la variabile d'ambiente **`LICENSE`**.

Variabili d'ambiente supportate

LANG
Impostare la lingua in cui si desidera stampare la licenza.

LICENSE
Impostare `accept` per accettare le condizioni della licenza IBM MQ Advanced for Developers.

Impostare `view` per visualizzare le condizioni di licenza.

MQ_CMDLEVEL
Imposta il livello di comando MQ da utilizzare.

MQ_ENABLE_EMBEDDED_WEB_SERVER
Impostare su `true` per avviare il server web all'avvio del contenitore.

MQ_ENABLE_FIPS
Impostare su `true` per abilitare la modalità FIPS . Impostare su `false` per disabilitare la modalità FIPS . Per impostazione predefinita, la modalità FIPS viene abilitata automaticamente se l'host Linux sottostante ha abilitato FIPS , ma questa opzione può essere utile per i test.

MQ_ENABLE_METRICS
Impostare `true` per generare le metriche di Prometheus per il gestore di code.

MQ_ENABLE_TRACE_CRTMQDIR
Impostare su "true" per creare una traccia diagnostica MQ durante l'esecuzione del comando **`crtmqdir`**.

MQ_ENABLE_TRACE_CRTMQM
Impostare su "true" per creare una traccia diagnostica MQ durante l'esecuzione del comando **`crtmqm`**.

MQ_ENABLE_TRACE_STRMQM
Impostare su "true" per creare una traccia diagnostica MQ durante l'esecuzione del comando **`strmqm`**.

MQ_GRAZIA_PERIODO
Il tempo target in cui si tenta di terminare il gestore di code all'interno, intensificando le fasi di disconnessione dell'applicazione. Imposta l'opzione **`endmqm -tp`** utilizzata quando il contenitore termina. Per ulteriori informazioni, vedere endmqm (end queue manager).

MQ_LOGGING_CONSOLE_SOURCE
Specificare un elenco separato da virgole di sorgenti per i registri che vengono rispecchiati nella posizione **`stdout`** del contenitore.

I valori validi sono `qmgr`, `web` e `mqsc`.

Il valore predefinito è `qmgr,web`.

Il valore opzionale è `mqsc`. Questa opzione può essere usata per riflettere il contenuto di `autocfgmqsc.LOG` nel log del contenitore.

MQ_LOGGING_CONSOLE_FORMAT
Cambia il formato dei registri che vengono stampati nella posizione **`stdout`** del contenitore.

Impostare `basic` per utilizzare un formato semplice e leggibile. Questo è il valore predefinito.

Impostare `json` per usare il formato JSON (un oggetto JSON su ogni riga).

MQ_LOGGING_CONSOLE_EXCLUDE_ID
Specificare un elenco separato da virgole di ID messaggio per i messaggi di registro da escludere.

I messaggi di log appaiono ancora nel file di log su disco, ma non vengono stampati nella posizione **`stdout`** del contenitore.

Il valore predefinito è `AMQ5041I,AMQ5052I,AMQ5051I,AMQ5037I,AMQ5975I`.

V 9.4.3 **MQ_LOGGING_METRICS_AUDIT_ENABLED**
Impostare su "true" per abilitare la registrazione degli accessi all'endpoint delle metriche di Prometheus. L'output del log viene inviato a un file JSON in `/var/mqm/errors/`. È necessario impostare anche `MQ_ENABLE_METRICS=true`, per generare le metriche di Prometheus.

MQ_MULTI_INSTANCE
Impostare "true" per abilitare l'esecuzione come gestore di code a più istanze. Questo cambia le opzioni utilizzate con **endmqm**.

MQ_NATIVO_HA
Impostare su "true" per abilitare l'HA nativo. Imposta l'opzione **crtmqm -lr**. Per ulteriori informazioni, vedere crtmqm (create queue manager). È possibile configurare altre impostazioni di Native HA montando un file INI. Vedere la stanza 'NativeHAInstance del file qm.ini e la stanza 'NativeHALocalInstance del file qm.ini.

Deprecated **MQ_NATIVE_HA_INSTANCE_0_NAME**
Imposta l'attributo 'Name nella stanza INI 'NativeHAInstance per una delle tre istanze Native HA. Vedere la stanza 'NativeHAInstance del file qm.ini Da 'IBM MQ 9.4.1, questa variabile è deprecata e si deve invece fornire un frammento di file INI in '/etc/mqm.

Deprecated **MQ_NATIVE_HA_INSTANCE_1_NAME**
Imposta l'attributo 'Name nella stanza INI 'NativeHAInstance per una delle tre istanze Native HA. Vedere la stanza 'NativeHAInstance del file qm.ini Da 'IBM MQ 9.4.1, questa variabile è deprecata e si deve invece fornire un frammento di file INI in '/etc/mqm.

Deprecated **MQ_NATIVE_HA_INSTANCE_2_NAME**
Imposta l'attributo 'Name nella stanza INI 'NativeHAInstance per una delle tre istanze Native HA. Vedere la stanza 'NativeHAInstance del file qm.ini Da 'IBM MQ 9.4.1, questa variabile è deprecata e si deve invece fornire un frammento di file INI in '/etc/mqm.

Deprecated **MQ_NATIVE_HA_INSTANCE_0_REPLICATION_ADDRESS**
Imposta l'attributo 'ReplicationAddress nella stanza INI 'NativeHAInstance per una delle tre istanze Native HA. Vedere la stanza 'NativeHAInstance del file qm.ini Da 'IBM MQ 9.4.1, questa variabile è deprecata e si deve invece fornire un frammento di file INI in '/etc/mqm.

Deprecated **MQ_NATIVE_HA_INSTANCE_1_REPLICATION_ADDRESS**
Imposta l'attributo 'ReplicationAddress nella stanza INI 'NativeHAInstance per una delle tre istanze Native HA. Vedere la stanza 'NativeHAInstance del file qm.ini Da 'IBM MQ 9.4.1, questa variabile è deprecata e si deve invece fornire un frammento di file INI in '/etc/mqm.

Deprecated **MQ_NATIVE_HA_INSTANCE_2_REPLICATION_ADDRESS**
Imposta l'attributo 'ReplicationAddress nella stanza INI 'NativeHAInstance per una delle tre istanze Native HA. Vedere la stanza 'NativeHAInstance del file qm.ini Da 'IBM MQ 9.4.1, questa variabile è deprecata e si deve invece fornire un frammento di file INI in '/etc/mqm.

Deprecated **MQ_NATIVE_HA_CIPHERSPEC**
Imposta l'attributo 'CipherSpec nella stanza INI 'NativeHALocalInstance per una delle tre istanze Native HA. Vedere la stanza 'NativeHALocalInstance del file qm.ini Da 'IBM MQ 9.4.1, questa variabile è deprecata e si deve invece fornire un frammento di file INI in '/etc/mqm.

Deprecated **MQ_NATIVE_HA_KEY_REPOSITORY**
Ignora il repository di chiavi generato automaticamente per Native HA e utilizza quello specificato. Da 'IBM MQ 9.4.1, questa variabile è deprecata e si deve invece fornire la posizione del repository delle chiavi in un frammento di file INI in '/etc/mqm.

MQ_QMGR_LOG_FILE_PAGES
MQ sono conservati in una serie di file chiamati file di registro. La dimensione del file di registro è specificata in unità di pagine da 4 KB. Questa variabile d'ambiente imposta l'opzione **crtmqm -lf**. Per ulteriori informazioni, vedere crtmqm (create queue manager).

MQ_QMGR_NAME
Impostare il nome con cui si vuole creare il gestore di code.

Sistema di file e punti di montaggio

/etc/mqm
Qualsiasi file MQSC o INI presente in questa directory verrà elaborato dal gestore di code all'avvio. È la directory utilizzata dalla funzione di configurazione automatica. Per ulteriori informazioni, vedere Configurazione automatica da uno script MQSC all'avvio e Configurazione automatica di qm.ini all'avvio.

/etc/mqm/pki/keys
Contiene sottodirectory con chiavi private in formato PKCS #1, ASN.1 DER, o PKCS #8, ASN.1 DER, non criptate. Ogni sottodirectory sotto `/etc/mqm/pki/keys` viene analizzata alla ricerca di qualsiasi file con estensione `.key`. La catena di certificati pubblici X.509 deve trovarsi nella stessa directory, in uno o più file con estensione " `.crt`, in formato ASN.1 DER. Se le chiavi vengono montate in questo modo, viene generato automaticamente un deposito di chiavi in '`/run/runmqserver/tls`. Il repository delle chiavi sarà impostato nell'impostazione " SSLKEYR del gestore di code e la chiave predefinita (CERTLABL) sarà il primo nome dopo l'ordinamento lessicografico dell'elenco dei nomi delle chiavi.

/etc/mqm/pki/trust
Contiene sottodirectory con certificati pubblici X.509 in forma ASN.1 DER. Ogni sottodirectory verrà analizzata alla ricerca di eventuali file `.crt` e aggiunta come certificato attendibile. I certificati delle chiavi private sono automaticamente attendibili e non è necessario aggiungerli in questo modo.

/etc/mqm/ha/pki/keys
Contiene sottodirectory con chiavi private in formato PKCS #1, ASN.1 DER, o PKCS #8, ASN.1 DER, non criptate. Ogni sottodirectory sotto `/etc/mqm/ha/pki/keys` viene analizzata alla ricerca di qualsiasi file con estensione `.key`. La catena di certificati pubblici X.509 deve trovarsi nella stessa directory, in uno o più file con estensione " `.crt`, in formato ASN.1 DER. Se le chiavi vengono montate in questo modo, viene generato automaticamente un deposito di chiavi e un frammento di file INI per utilizzarlo. La chiave utilizzata sarà il primo nome dopo l'ordinamento lessicografico dell'elenco dei nomi delle chiavi. Il repository di chiavi generato viene scritto in '`/run/runmqserver/ha/tls`.

/etc/mqm/metrics/pki/keys
Da 9.4.2.0-r1, se in questa directory sono presenti chiavi TLS e le metriche sono abilitate per un gestore di code, viene avviato un server HTTPS sulla porta 9157 di `/metrics` . I file in questa directory devono essere certificati con codifica PEM:

- `tls.crt` certificato pubblico del server
- `tls.key` Chiave privata del server
- `ca.crt` Certificato pubblico CA (opzionale)

Se non vengono fornite chiavi TLS o se vengono utilizzate versioni precedenti di IBM MQ , viene avviato un server HTTP.

/etc/mqm/web
L'albero delle cartelle sotto `/etc/mqm/web` viene copiato sopra `/var/mqm/web` all'avvio del contenitore. Ad esempio, si può creare `/etc/mqm/web/installations/Installation1/servers/mqweb/mqwebuser.xml` per configurare il server web. Questo file viene copiato nella posizione giusta sotto il " `/var/mqm/web`, che di solito è un volume montato. Vedere Configurazione di 'IBM MQ Console e 'REST API.

/mnt/mqm
La directory `/var/mqm` è simbolicamente collegata a questa posizione. Montando un volume persistente in questa posizione, è possibile persistere i dati MQ tra le esecuzioni del contenitore.

/mnt/mqm-log
Directory utilizzata per contenere i file di registro di recupero per il gestore di code. Se un volume viene montato in questa posizione, il codice del contenitore imposta l'opzione **`crtmqm -ld`**. Per ulteriori informazioni, vedere crtmqm (create queue manager).

/mnt/mqm-data
Directory utilizzata per contenere i file di dati per il gestore di code. Se un volume viene montato in questa posizione, il codice del contenitore imposta l'opzione **crtmqm -md**. Per ulteriori informazioni, vedere crtmqm (create queue manager).

/run
Il contenitore scrive i file temporanei in questa directory. Se si vuole utilizzare un filesystem di sola lettura per il contenitore, è necessario montare un volume in questa posizione e in `/tmp`

/run/termination-log
Se il codice del contenitore incontra un errore che causa l'interruzione del gestore di code, scriverà il motivo in questo file. Questa posizione del file viene utilizzata per impostazione predefinita in Kubernetes per recuperare le informazioni sulla terminazione.

/tmp
Il contenitore scrive i file temporanei in questa directory. Se si vuole utilizzare un filesystem di sola lettura per il contenitore, è necessario montare un volume in questa posizione e in `/run`

Linux IBM MQ immagine del contenitore con licenza

Per IBM MQ è disponibile un'immagine del contenitore precostruita. Questa immagine è disponibile sul sito IBM Container Registry.

Utilizzo

Per poter utilizzare l'immagine, è necessario accettare i termini della licenza IBM MQ impostando la variabile d'ambiente **LICENSE** . Per ulteriori informazioni, consultare “Configurare i gestori di code con le annotazioni di licenza di IBM MQ utilizzando il file IBM MQ Operator” a pagina 160.

Variabili d'ambiente supportate

Le variabili d'ambiente supportate dalle immagini del contenitore con licenza IBM MQ sono le stesse di quelle supportate dall'immagine del contenitore IBM MQ Advanced . Per un elenco completo, vedere Variabili d'ambiente supportate dall'immagine del contenitore IBM MQ Advanced.

Nota: Da IBM MQ 9.4.2, IBM MQ è possibile configurare i gestori di code con licenza per utilizzare le funzionalità Native HA e Cross-Region Replication aggiungendo licenze supplementari a un gestore di code. Per ulteriori informazioni, consultare “Configurazione dell'HA nativo e della replica interregionale sui gestori di code con licenza IBM MQ utilizzando il software IBM MQ Operator” a pagina 115. Le funzionalità HA native non possono essere utilizzate su gestori di code precedenti a IBM MQ 9.4.2 che utilizzano una licenza IBM MQ .

File system e punti di montaggio

Il file system e i punti di mount supportati dalle immagini container con licenza IBM MQ sono gli stessi supportati dall'immagine container IBM MQ Advanced . Per un elenco completo, vedere File system e punti di mount nell'immagine del contenitore IBM MQ Advanced.

OpenShift CP4I Kubernetes IBM MQ Advanced for Developers immagine contenitore

Un'immagine del contenitore precostruita è disponibile per IBM MQ Advanced for Developers. Questa immagine è disponibile da IBM Container Registry. Questa immagine è adatta all'uso con Docker, Podman, Kubernetes e altri ambienti container.

Immagini disponibili

Le immagini IBM MQ sono memorizzate in IBM Container Registry:

- IBM MQ Advanced for Developers 9.4.0.11: `icr.io/ibm-messaging/mq:9.4.0.11-r3`

- IBM MQ Advanced for Developers 9.4.3.0: `icr.io/ibm-messaging/mq:9.4.3.0-r1`

Riferimento rapido

- Licenza:
  - “mq.ibm.com/v1beta1: Versioni di licenza attuali” a pagina 248 e Apache Licenza 2.0. Tenere presente che la licenza IBM MQ Advanced for Developers non consente un'ulteriore distribuzione e che i termini limitano l'uso a una macchina dello sviluppatore.
- Dove archiviare i problemi:
  - GitHub
- Disponibile per le seguenti architetture CPU:
  - amd64
  - s390x
  - ppc64le

Utilizzo

Esegui IBM MQ Advanced for Developers in un contenitore.

Vedi la documentazione sull'utilizzo per i dettagli su come eseguire un contenitore.

Per poter utilizzare l'immagine, devi accettare i termini della licenza IBM MQ impostando la variabile di ambiente **LICENSE** .

Variabili di ambiente

Sono disponibili tutte le variabili d'ambiente di “IBM MQ Advanced immagine contenitore” a pagina 287, più le seguenti variabili aggiuntive:

Deprecated **MQ_ADMIN_PASSWORD**
Specificare la password dell'utente admin.

Deve contenere almeno 8 caratteri.

Non esiste alcuna password predefinita per l'utente admin.

V 9.4.0 V 9.4.0 Da IBM MQ 9.4.0, questa variabile non è più fornita. L 'esempio YAML in questo argomento mostra come si possa fornire la password di amministrazione montando un segreto.

Deprecated **MQ_APP_PASSWORD**
Specificare la password dell'utente dell'app.

Se impostato, questo fa sì che il canale **DEV.APP.SVRCONN** diventi protetto e consenta solo le connessioni che forniscono un ID utente e una password validi.

Deve contenere almeno 8 caratteri.

Non esiste alcuna password predefinita per l'utente dell'applicazione.

V 9.4.0 V 9.4.0 Da IBM MQ 9.4.0, questa variabile non è più fornita. L'esempio di YAML in questo argomento mostra come sia possibile fornire la password dell'applicazione montando un segreto....

MQ_DEV
Impostare `false` per interrompere la creazione degli oggetti predefiniti.

Per ulteriori informazioni sulla configurazione dello sviluppatore predefinito supportata dall'immagine IBM MQ Advanced for Developers , vedi la documentazione della configurazione dello sviluppatore predefinito.

YAML del gestore code di esempio che descrive come specificare le password per gli utenti admin e app

Per gli utenti degli ID utente **admin** e **app** , è necessario fornire le password durante la distribuzione di un gestore code utilizzando la licenza `Development` . Di seguito è riportato un esempio di YAML del gestore code che mostra come eseguire questa operazione con IBM MQ Operator.

Il comando riportato di seguito crea un segreto contenente le password per gli utenti **admin** e **app** .

```
oc create secret generic my-mq-dev-passwords --from-literal=mqAdminPassword=passw0rd --from-
literal=mqAppPassword=passw0rd
```

Il seguente YAML utilizza tali password quando si distribuisce un gestore code.

```
apiVersion: mq.ibm.com/v1beta1
kind: QueueManager
metadata:
  name: qm-dev
  annotations: com.ibm.mq/write-defaults
  spec: 'false'
spec:
  license:
    accept: true
    license: L-HYGL-6STWD6
    use: Development
  web:
    enabled: true
  queueManager:
    env:
      - name: MQ_DEV
        value: "true"
      - name: MQ_CONNAUTH_USE_HTP
        value: "true"
    files:
      - secret:
          items:
            - item: mqAdminPassword
            - item: mqAppPassword
      name: my-mq-dev-passwords
      defaultMountPath: /run/secrets
    storage:
      queueManager:
        type: persistent-claim
    name: QUICKSTART
  version: 9.4.3.0-r1
```

IBM MQ MFT Agent for Developers immagine contenitore

Per IBM MQ MFT Agent for Developers è disponibile un'immagine del contenitore precostruita. Questa immagine è disponibile sul sito IBM Container Registry. Questa immagine è adatta all'uso con Docker, Podman, Kubernetes e altri ambienti container.

Attenzione: Deprecated IBM MQ MFT Agent for Developers le immagini erano precedentemente disponibili su Docker Hub, ma questa funzione è stata abbandonata e non sono disponibili ulteriori aggiornamenti su Docker Hub.

Immagini disponibili

IBM MQ MFT Agent for Developers le immagini sono memorizzate nel sito IBM Container Registry:

- IBM MQ 9.3.0.0: `icr.io/ibm-messaging/mqmft:latest`

Riferimento rapido

- Licenza:
  - IBM MQ Advanced for Developers e Apache Licenza 2.0. La licenza IBM MQ Advanced for Developers non consente ulteriori distribuzioni e i termini limitano l'uso a una macchina per sviluppatori.

- Dove presentare i problemi:
  - GitHub
- Disponibile per le seguenti architetture di CPU:
  - amd64

Creare l'immagine di un contenitore

Consultare la documentazione d'uso per creare l'immagine del proprio contenitore agente con il pacchetto MQ Managed File Transfer Redistributable su architetture amd64..

Utilizzo

Vedere la documentazione d'uso per i dettagli su come eseguire il contenitore di immagini con il runtime Podman.

Per i dettagli su come distribuire l'immagine in un sito OpenShift Container Platform, consultare la documentazione d'uso.

Per poter utilizzare l'immagine, è necessario accettare i termini della licenza IBM MQ impostando la variabile d'ambiente **LICENSE** .

Variabili di ambiente

LICENSE
Impostate `Accetta` per accettare le condizioni di licenza di IBM MQ Advanced for Developers .

Impostare la `vista` per visualizzare le condizioni di licenza.

MFT_AGENT_CONFIG_FILE
Obbligatorio.

Percorso del file json contenente le informazioni necessarie per la configurazione di un agente. Il percorso deve trovarsi in un punto di montaggio. Ad esempio un `configMap` su OpenShift. Per una descrizione dettagliata degli attributi, consultare i file di configurazione dell'agente Additional MFT .

MFT_AGENT_NAME
Specificare la password dell'utente admin.

Nome dell'agente da configurare.

BFG_JVM_PROPRIETÀ
Facoltativo.

Qualsiasi proprietà della JVM che deve essere impostata durante l'esecuzione della JVM agente.

MFT_LOG_LEVEL
Facoltativo.

Livello di informazioni visualizzate. `info` e `verbose` sono i valori supportati, con `info` come *valore predefinito*. Il contenuto di `output0.log` dell'agente viene visualizzato se MFT_LOG_LEVEL è impostato su `verbose`.

MFT_AGENT_START_WAIT_TIME
Facoltativo.

L'avvio di un agente potrebbe richiedere un certo tempo dopo l'emissione del comando **`fteStartAgent`** . È il tempo, in secondi, in cui il contenitore attende l'avvio di un agente. Se un agente non lo fa entro il tempo di attesa specificato, il contenitore viene terminato.

MFT_MOUNT_PATH
Facoltativo.

Variabile d'ambiente che punta al percorso da cui l'agente leggerà i file o su cui scriverà.

MFT_COORD_QMGR_CIPHER
Il nome di CipherSpec da utilizzare per la connessione sicura al gestore di code di coordinamento.

MFT_CMD_QMGR_CIPHER
Il nome di CipherSpec da utilizzare per la connessione sicura al gestore delle code di comando.

MFT_AGENT_QMGR_CIPHER
Il nome di CipherSpec da usare per connettersi in modo sicuro a Agent Queue Manager.

Per ulteriori informazioni sulla configurazione predefinita dello sviluppatore supportata da IBM MQ Advanced for Developers immagine, vedere la documentazione sulla configurazione predefinita dello sviluppatore.

Posizione dei file di configurazione dell'agente

L'agente nel contenitore creerà i file di configurazione e di log dell'agente nella directory fissa `/mnt/mftdata`. Questa cartella può essere anche su un volume persistente, nel qual caso il volume deve essere montato come punto di montaggio `/mnt/mftdata` nel contenitore.

OpenShift CP4I IBM MQ CASE per l'installazione e l'aggiornamento air-gap su 'Red Hat OpenShift

Quando si installano i gestori di code 'IBM MQ Operator e 'IBM MQ in un cluster 'Red Hat OpenShift con air-gapped, si utilizza IBM MQ CASE. IBM MQ CASE contiene i dettagli di tutte le immagini e i metadati che compongono una particolare release di IBM MQ CASE.

Air-gap e IBM MQ CASE 3.5.0 modifiche

Da IBM MQ CASE 3.5.0 in poi, IBM MQ CASE contiene immagini per una sola versione del gestore di code. Questa è l'ultima versione del gestore di code che il corrispondente IBM MQ Operator può distribuire. Prima di IBM MQ CASE 3.5.0, CASE conteneva tutte le versioni di queue manager che il gestore corrispondente poteva distribuire. Vedi l' IBM MQ CASE Lookup Tables for the exact version available in each IBM MQ CASE.

È possibile utilizzare un IBM MQ CASE per installare, o aggiornare, l'ultimo gestore di code disponibile che il gestore può distribuire, ovvero il gestore di code CD o SC2 rilasciato contemporaneamente al gestore CASE e al gestore corrispondente. In questo modo, il processo di installazione o di aggiornamento rimane lo stesso ma viene eseguito più rapidamente, perché deve rispecchiare un numero inferiore di immagini.

Tuttavia, se si installa una o più versioni del gestore di code che non sono le più recenti che l'operatore può distribuire, è necessario ripetere il processo di mirroring dell'immagine air-gap utilizzando un IBM MQ CASE separato per ciascuna versione del gestore di code richiesta. Vedi l' IBM MQ CASE Lookup Tables for the exact version available in each IBM MQ CASE.

Nota: IBM MQ Operator , che è ancora a conoscenza delle versioni precedenti di Queue Manager, può quindi essere ancora utilizzato per implementare versioni precedenti di Queue Manager. Questo è il motivo per cui è importante considerare quali versioni di queue manager è necessario rispecchiare.

Immagini relative al gestore delle code

Da IBM MQ CASE 3.5.0 in poi, IBM MQ CASE contiene solo le immagini per la singola versione più recente di queue manager che il gestore corrispondente può distribuire. In precedenza, il sito CASE conteneva tutte le versioni di queue manager che il gestore corrispondente poteva distribuire.

Nota: Da 3.5.0, IBM MQ CASE non sono più incluse le immagini del pannello operativo, perché non è un'opzione per le ultime versioni del gestore di code per implementare il pannello operativo.

Use the IBM MQ CASE Lookup Tables to identify which IBM MQ CASE contains the IBM MQ queue manager version you require.

Immagini relative all'operatore

Utilizza le tabelle di ricerca IBM MQ CASE per identificare quale IBM MQ CASE contiene la versione IBM MQ Operator che ti serve.

Importante:

- Quando si installa IBM MQ Operator in un ambiente air-gap, si deve sempre completare il processo di mirroring air-gap per IBM MQ CASE che corrisponde alla versione di IBM MQ Operator che si intende installare. Non installare o aggiornare a una versione operatore da un altro IBM MQ CASE.
- IBM MQ CASE contiene ulteriori versioni di IBM MQ Operator, ma installate solo l'operatore corrispondente a IBM MQ CASE che avete copiato.
- Quando si aggiorna IBM MQ Operator, se si cambia canale operatore, seguire la best practice di aggiornamento air-gap per aggiornare alla versione più recente di IBM MQ Operator disponibile sul canale corrente. È importante farlo prima di cambiare l'abbonamento a un nuovo canale IBM MQ Operator.

IBM MQ CASE Tabelle di ricerca

Tabella 3. Contenuto dei CD recenti di IBM MQ CASE

IBM MQ CASE	IBM MQ Operatore	Gestore code IBM MQ
3.6.0	3.6.0	9.4.3.0-r1
3.5.3	3.5.3	9.4.2.1-r2
3.5.2	3.5.2	9.4.2.1-r1
3.5.1	3.5.1	9.4.2.0-r2
3.5.0	3.5.0	9.4.2.0-r1
3.4.1	3.4.1	9.4.1.1-r1, 9.4.1.0-r2, 9.4.1.0-r1, 9.4.0.7-r1, 9.4.0.6-r2, 9.4.0.6-r1, 9.4.0.5-r2, 9.4.0.5-r1, 9.4.0.0-r3, 9.4.0.0-r2, 9.4.0.0-r1, 9.3.5.1-r2, 9.3.5.1-r1, 9.3.5.0-r2, 9.3.5.0-r1, 9.3.4.1-r1, 9.3.4.0-r1, 9.3.3.3-r2, 9.3.3.3-r1, 9.3.3.2-r3, 9.3.3.2-r2, 9.3.3.2-r1, 9.3.3.1-r2, 9.3.3.1-r1, 9.3.3.0-r2, 9.3.3.0-r1, 9.3.2.1-r2, 9.3.2.1-r1, 9.3.2.0-r2, 9.3.2.0-r1, 9.3.1.1-r1, 9.3.1.0-r3, 9.3.1.0-r2, 9.3.1.0-r1, 9.3.0.25-r1, 9.3.0.21-r3, 9.3.0.21-r2, 9.3.0.21-r1, 9.3.0.20-r2, 9.3.0.20-r1, 9.3.0.17-r3, 9.3.0.17-r2, 9.3.0.17-r1, 9.3.0.16-r2, 9.3.0.16-r1, 9.3.0.15-r1, 9.3.0.11-r2, 9.3.0.11-r1, 9.3.0.10-r2, 9.3.0.10-r1, 9.3.0.6-r1, 9.3.0.5-r3, 9.3.0.5-r2, 9.3.0.5-r1, 9.3.0.4-r2, 9.3.0.4-r1, 9.3.0.3-r1, 9.3.0.1-r4, 9.3.0.1-r3, 9.3.0.1-r2, 9.3.0.1-r1, 9.3.0.0-r3, 9.3.0.0-r2, 9.3.0.0-r1

Tabella 4. Contenuto delle istanze recenti di SC2 IBM MQ CASE		
IBM MQ CASE	**IBM MQ Operatore**	**Gestore code IBM MQ**
3.2.13	3.2.13	9.4.0.11-r3, 9.4.0.11-r2, 9.4.0.11-r1, 9.4.0.10-r2, 9.4.0.10-r1, 9.4.0.7-r1, 9.4.0.6-r2, 9.4.0.6-r1, 9.4.0.5-r2, 9.4.0.5-r1, 9.4.0.0-r3, 9.4.0.0-r2, 9.4.0.0-r1, 9.3.5.1-r2, 9.3.5.1-r1, 9.3.5.0-r2, 9.3.5.0-r1, 9.3.4.1-r1, 9.3.4.0-r1, 9.3.3.3-r2, 9.3.3.3-r1, 9.3.3.2-r3, 9.3.3.2-r2, 9.3.3.2-r1, 9.3.3.1-r2, 9.3.3.1-r1, 9.3.3.0-r2, 9.3.3.0-r1, 9.3.2.1-r2, 9.3.2.1-r1, 9.3.2.0-r2, 9.3.2.0-r1, 9.3.1.1-r1, 9.3.1.0-r3, 9.3.1.0-r2, 9.3.1.0-r1, 9.3.0.25-r1, 9.3.0.21-r3, 9.3.0.21-r2, 9.3.0.21-r1, 9.3.0.20-r2, 9.3.0.20-r1, 9.3.0.17-r3, 9.3.0.17-r2, 9.3.0.17-r1, 9.3.0.16-r2, 9.3.0.16-r1, 9.3.0.15-r1, 9.3.0.11-r2, 9.3.0.11-r1, 9.3.0.10-r2, 9.3.0.10-r1, 9.3.0.6-r1, 9.3.0.5-r3, 9.3.0.5-r2, 9.3.0.5-r1, 9.3.0.4-r2, 9.3.0.4-r1, 9.3.0.3-r1, 9.3.0.1-r4, 9.3.0.1-r3, 9.3.0.1-r2, 9.3.0.1-r1, 9.3.0.0-r3, 9.3.0.0-r2, 9.3.0.0-r1

Rimozione dell'operatore precedente

Per riferimento, da IBM MQ CASE 3.4.0, le immagini IBM MQ Operator per i seguenti canali sono state rimosse da IBM MQ CASE :

- v1.1
- v1.2
- v1.4
- v1.5
- v1.6
- v1.7
- v2.1

Ricreare cluster e registri immagini

È possibile utilizzare il seguente comando per elencare tutti i gestori di code distribuiti in un cluster di Red Hat OpenShift , e le loro versioni.

```
oc get queuemanager --all-namespaces -o=jsonpath='{range .items[*]}{.metadata.name}{":\t"}
{.spec.version}{"\n"}{end}'
```

È necessario eseguire questo comando periodicamente e memorizzare l'output. Questo elenco è utile se si desidera ricreare il cluster dell' Red Hat OpenShift , o il Registro immagini che memorizza le immagini replicate per l'air-gap.

Attività correlate

“Aggiornamento di un canale 'CD 'IBM MQ Operator ' al canale 'CD attuale” a pagina 76
Aggiornare una distribuzione 'CD di 'IBM MQ Operator in cui vengono utilizzate **solo le** licenze 'IBM MQ.

“Aggiornamento di un 'SC2 'IBM MQ Operator '3.2.x al canale 'CD ' attuale” a pagina 75
Aggiornare una distribuzione 3.2.x 'SC2 di 'IBM MQ Operator in cui sono utilizzate **solo** licenze 'IBM MQ.

“Installazione del IBM MQ Operator” a pagina 58
Il IBM MQ Operator può essere installato su Red Hat OpenShift utilizzando la console OpenShift o la CLI (command line interface).

“Installazione di IBM MQ Operator per l'utilizzo con CP4I” a pagina 65
Per l'utilizzo con IBM Cloud Pak for Integration (CP4I), IBM MQ Operator può essere installato su Red Hat OpenShift tramite la console OpenShift o la CLI (command line interface).

Informazioni particolari

Queste informazioni sono state sviluppate per prodotti e 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. Tuttavia, è responsabilità dell'utente valutare e verificare il funzionamento di qualsiasi prodotto, programma o servizio non 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:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
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
Legal and Intellectual Property Law
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.

Tutti i riferimenti a siti Web non dell'IBM contenuti in questo documento sono forniti solo per consultazione e non rappresenta in alcun modo un'approvazione di tali siti. I materiali reperibili in tali siti Web non fanno parte dei materiali relativi a questo prodotto IBM e l'utilizzo di tali siti è responsabilità dell'utente.

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 base ai termini dell' 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 poterli illustrare nel modo più completo possibile, gli esempi riportano nomi di persone, 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 IBM 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 poiché 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, in altri paesi o entrambi.

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: