aws

Guida per gli sviluppatori

Amazon Simple Queue Service

Amazon Simple Queue Service: Guida per gli sviluppatori

Table of Contents

Che cos'è Amazon Simple Queue Service?

Amazon Simple Queue Service (Amazon SQS) offre una coda ospitata internamente sicura, durevole e disponibile che consente di integrare e separare i componenti e i sistemi software distribuiti. Amazon SQS offre costrutti comuni come, ad esempio, code DLQ e tag di allocazione dei costi. Fornisce anche un'API web service generica e puoi accedervi mediante qualsiasi linguaggio di programmazione supportato dall'SDK AWS.

Argomenti

Vantaggi dell'utilizzo di Amazon SQS

- Sicurezza - Puoi controllare chi può inviare e ricevere messaggi da una coda Amazon SQS. Puoi scegliere di trasmettere dati sensibili proteggendo il contenuto dei messaggi nelle code utilizzando la crittografia lato server gestita (SSE) predefinita di Amazon SQS o utilizzando chiavi SSE personalizzate gestite in AWS Key Management Service (AWS KMS).
- Durabilità: per garantire la sicurezza dei tuoi messaggi, Amazon SQS li memorizza su più server. Le code standard supportano il recapito dei at-least-once messaggi, mentre le code FIFO supportano l'elaborazione dei messaggi una sola volta e la modalità high-throughput.
- Disponibilità: Amazon SQS utilizza un'infrastruttura ridondante per fornire l'accesso simultaneo ai messaggi e alta disponibilità per la produzione e l'utilizzo di messaggi.
- Scalabilità: Amazon SQS è in grado di elaborare ogni richiesta di buffer in modo indipendente e di ridimensionarsi in maniera trasparente per gestire eventuali picchi o aumenti di carico senza le relative istruzioni di provisioning.
- Affidabilità: Amazon SQS blocca i tuoi messaggi durante l'elaborazione, in modo tale che più produttori possono inviare e più consumatori possono ricevere messaggi contemporaneamente.
- Personalizzazione: le tue code non devono essere esattamente uguali, per esempio, è possibile impostare un intervallo predefinito su una coda. Puoi archiviare i contenuti dei messaggi di dimensioni superiori a 256 KB utilizzando Amazon Simple Storage Service (Amazon S3) o Amazon DynamoDB con Amazon SQS che mantiene un puntatore all'oggetto Amazon S3, oppure puoi dividere un messaggio di grandi dimensioni in messaggi più piccoli.

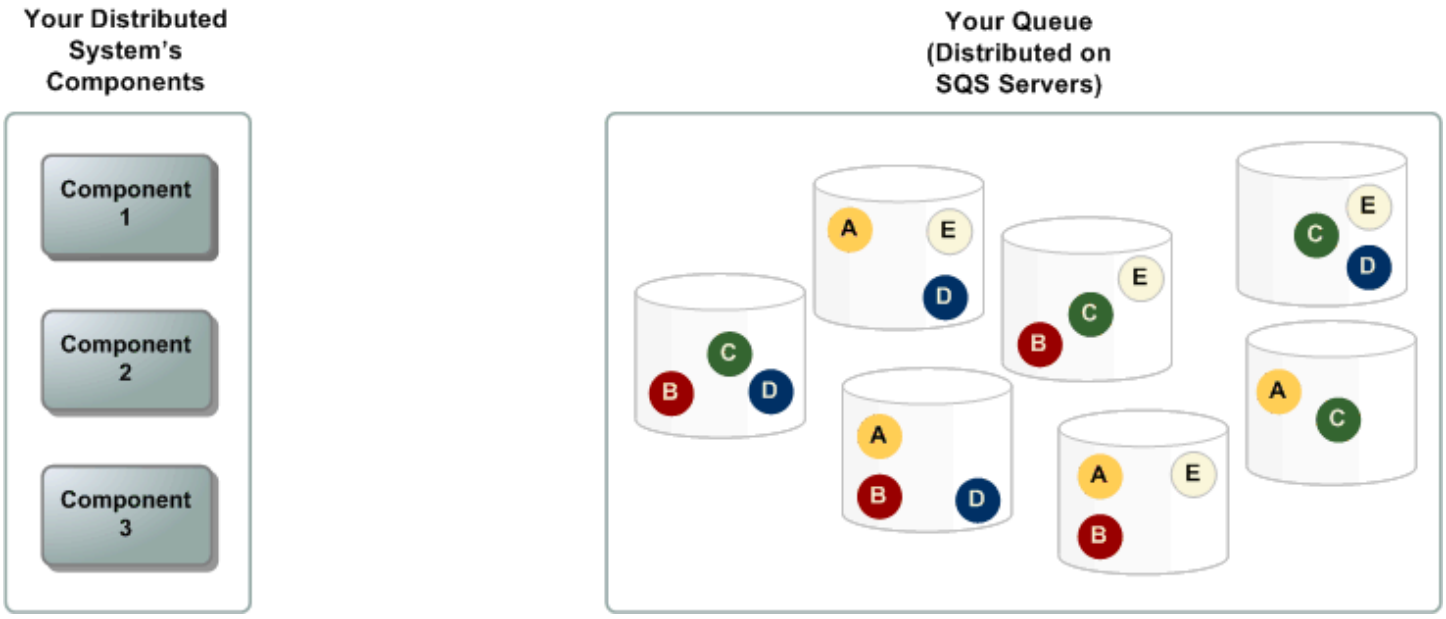
Architettura Amazon SQS di base

Questa sezione illustra le parti di un sistema di messaggistica distribuito e spiega il ciclo di vita di un messaggio Amazon SQS.

Code distribuite

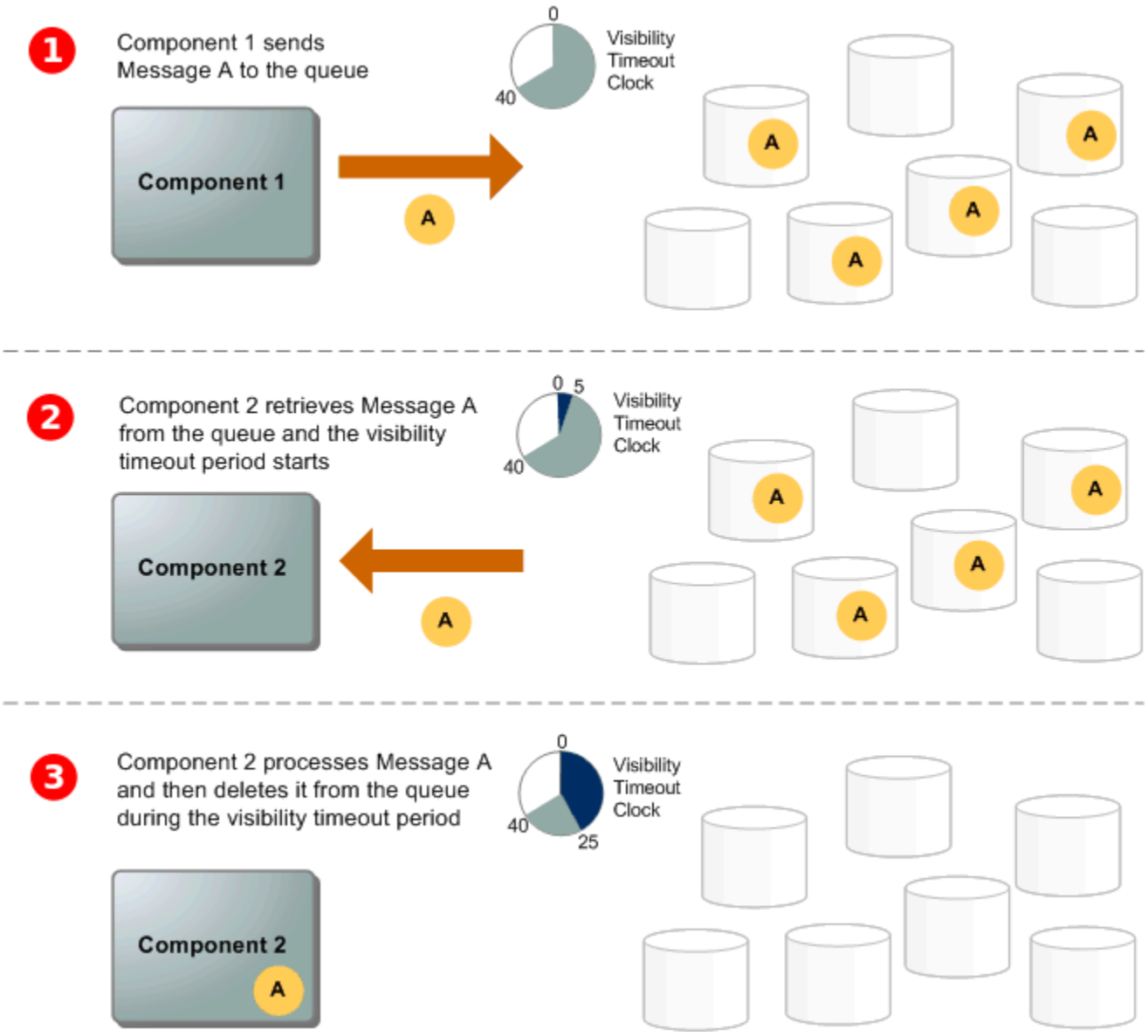
Esistono tre parti principali in un sistema di messaggistica distribuito: i componenti del sistema distribuito, la coda (distribuita su server Amazon SQS) e i messaggi nella coda.

Nel seguente scenario, il sistema dispone di diversi produttori (componenti che inviano messaggi alla coda) e consumatori (componenti che ricevono messaggi dalla coda). La coda (che contiene i messaggi da A a E) archivia in modo ridondante i messaggi su più server Amazon SQS.



Ciclo di vita del messaggio

Lo scenario seguente descrive il ciclo di vita di un messaggio Amazon SQS in una coda, dalla creazione all'eliminazione.



1

Un produttore (componente 1) invia il messaggio A a una coda e il messaggio viene distribuito in modo ridondante sui server SQS.

2

Quando un consumatore (componente 2) è pronto per elaborare i messaggi, utilizza i messaggi dalla coda e il messaggio A viene restituito. Mentre il messaggio A viene elaborato, rimane nella coda e non viene restituito alle richieste di ricezione successive per la durata del timeout visibilità.

3

Il consumatore (componente 2) elimina il messaggio A dalla coda per impedire che venga nuovamente ricevuto ed elaborato allo scadere del timeout visibilità.

> **Note**
>
> Amazon SQS elimina automaticamente i messaggi che sono stati in una coda per un periodo superiore quello massimo di conservazione. Il periodo predefinito per la conservazione dei messaggi è 4 giorni. Tuttavia, puoi impostare un valore compreso tra 60 secondi e 1.209.600 secondi (14 giorni) utilizzando l'azione `SetQueueAttributes` .

Differenze tra Amazon SQS, Amazon MQ e Amazon SNS

Amazon SQS, Amazon SNS e Amazon MQ sono servizi di messaggistica gestiti altamente scalabili e semplici da usare. Di seguito è riportata una panoramica delle differenze tra questi servizi:

Amazon SQS offre code ospitate che consentono di integrare e separare i componenti e i sistemi software distribuiti. Fornisce anche un'API web service generica a cui puoi accedervi mediante qualsiasi linguaggio di programmazione supportato dall'SDK AWS. I messaggi in coda vengono in genere elaborati da un singolo abbonato. Amazon SQS e Amazon SNS vengono spesso utilizzati insieme per creare un'applicazione di messaggistica fanout.

Amazon SNS è un servizio di pubblicazione e sottoscrizione che offre la consegna dei messaggi dagli editori (noti anche come produttori) a più endpoint di abbonati (noti anche come consumatori). Gli editori comunicano in modo asincrono con gli abbonati creando e inviando messaggi a un argomento, che rappresenta un punto di accesso logico e un canale di comunicazione. I clienti possono iscriversi all'argomento Amazon SNS e ricevere messaggi pubblicati utilizzando un tipo di endpoint supportato, come Amazon Data Firehose, Amazon SQS, Lambda, HTTP, email, notifiche push e messaggi di testo (SMS) per dispositivi mobili. Amazon SNS funge da router di messaggi e consegna i messaggi agli abbonati in tempo reale. Se un abbonato non è disponibile al momento della pubblicazione del messaggio, il messaggio non viene archiviato per un successivo recupero.

Amazon MQè un servizio di broker di messaggi gestito che offre compatibilità con i protocolli di messaggistica standard del settore come Advanced Message Queueing Protocol (AMQP) e Message Queuing Telemetry Transport (MQTT). Attualmente, Amazon MQ supporta i tipi di motore Apache ActiveMQ e RabbitMQ.

La tabella seguente fornisce una panoramica dei tipi di risorse di ciascun servizio:

Tipo di risorsa	Amazon SNS	Amazon SQS	Amazon MQ
Synchronous (Sincrona)	No	No	Sì
Asynchronous (Asincrona)	Sì	Sì	Sì
Queues	No	Sì	Sì
Messaggistica publish-subscribe	Sì	No	Sì
Broker di messaggi	No	No	Sì

Consigliamo Amazon SQS e Amazon SNS per le nuove applicazioni che possono sfruttare i vantaggi offerti da una scalabilità quasi illimitata e da semplici API. Consigliamo Amazon MQ per la migrazione di applicazioni da broker di messaggi esistenti che si basano sulla compatibilità con API come JMS o protocolli come Advanced Message Queuing Protocol (AMQP), MQTT e Simple Text Oriented Message Protocol (STOMP). OpenWire

Configurazione di Amazon SQS

Prima di utilizzare Amazon SQS per la prima volta, devi completare la procedura seguente.

Argomenti

Fase 1: creazione di un Account AWS e di un utente IAM

Per accedere a qualsiasi servizio AWS, devi prima creare un Account AWS, un account Amazon.com che può utilizzare i prodotti AWS. Puoi utilizzare il tuo Account AWS per visualizzare le attività e i report di utilizzo e per gestire l'autenticazione e l'accesso.

Per evitare di utilizzare l'account utente root Account AWS per le operazioni di Amazon SQS, è consigliabile creare un utente IAM per ogni persona che ha bisogno dell'accesso con privilegi di amministratore ad Amazon SQS.

Registrarsi per creare un Account AWS

Se non disponi di un Account AWS, completa la procedura seguente per crearne uno.

Per registrarsi a un Account AWS

1. Apri la pagina all'indirizzo https://portal.aws.amazon.com/billing/signup.
2. Segui le istruzioni online.

   Nel corso della procedura di registrazione riceverai una telefonata, durante la quale sarà necessario inserire un codice di verifica attraverso la tastiera del telefono.

   Durante la registrazione di un Account AWS, viene creato un Utente root dell'account AWS. L'utente root dispone dell'accesso a tutte le risorse e tutti i Servizi AWSnell'account. Come best practice di sicurezza, assegna l'accesso amministrativo a un utente amministrativo e utilizza solo l'utente root per eseguire attività che richiedono l'accesso di un utente root.

Al termine del processo di registrazione, riceverai un'e-mail di conferma da AWS. È possibile visualizzare l'attività corrente dell'account e gestire l'account in qualsiasi momento accedendo all'indirizzo https://aws.amazon.com/ e selezionando Il mio account.

Creazione di un utente amministratore

Dopo aver effettuato la registrazione di un Account AWS, proteggi Utente root dell'account AWS, abilita AWS IAM Identity Center e crea un utente amministratore in modo da non utilizzare l'utente root per le attività quotidiane.

Protezione dell'Utente root dell'account AWS

1. Accedi alla AWS Management Console come proprietario dell'account scegliendo Utente root e immettendo l'indirizzo email del Account AWS. Nella pagina successiva, inserisci la password.

   Per informazioni sull'accesso utilizzando un utente root, consulta la pagina Accesso come utente root della Guida per l'utente di Accedi ad AWS.
2. Abilita l'autenticazione a più fattori (MFA) per l'utente root.

   Per ricevere istruzioni, consulta Abilitazione di un dispositivo MFA virtuale per l'utente root dell'Account AWS (console) nella Guida per l'utente IAM.

Creazione di un utente amministratore

1. Abilita IAM Identity Center

   Per istruzioni, consulta Abilitazione di AWS IAM Identity Center nella Guida per l'utente di AWS IAM Identity Center.
2. In Centro identità AWS IAM, assegna l'accesso amministrativo a un utente amministrativo.

   Per un tutorial sull'utilizzo di IAM Identity Center directory come origine di identità, consulta Configure user access with the default IAM Identity Center directory nella Guida per l'utente di AWS IAM Identity Center.

Accesso come utente amministratore

- Per accedere con l'utente IAM Identity Center, utilizza l'URL di accesso che è stato inviato al tuo indirizzo e-mail quando hai creato l'utente IAM Identity Center.

Per informazioni sull'accesso utilizzando un utente IAM Identity Center, consulta Accedere al portale di accesso AWS nella Guida per l'utente Accedi ad AWS.

Fase 2: Concessione dell'accesso programmatico

Per utilizzare operazioni Amazon SQS (ad esempio, usando Java o tramite AWS Command Line Interface), devi disporre di un ID chiave di accesso e di una chiave di accesso segreta.

> **Note**
>
> L'ID chiave di accesso e la chiave di accesso segreta sono specifici per AWS Identity and Access Management. Non confonderli con le credenziali per altri servizi AWS, ad esempio coppie di chiavi Amazon EC2.

Gli utenti hanno bisogno di un accesso programmatico se desiderano interagire con AWS esternamente a AWS Management Console. La modalità con cui concedere l'accesso programmatico dipende dal tipo di utente che accede ad AWS.

Per fornire agli utenti l'accesso programmatico, scegli una delle seguenti opzioni.

Quale utente necessita dell'accesso programmatico?	Per	Come
Identità della forza lavoro (Utenti gestiti nel centro identità IAM)	Utilizza credenziali temporane e per firmare richieste programmatiche alla AWS CLI, agli SDK AWS o alle API AWS.	Segui le istruzioni per l'interfa ccia che desideri utilizzare. • Per la AWS CLI, consulta la pagina Configurazione della AWS CLI per l'uso di AWS IAM Identity Center nella Guida per l'utente dell'AWS Command Line Interface. • Per gli SDK AWS, gli strumenti e le API AWS, consulta la pagina Autentica

Quale utente necessita dell'accesso programmatico?	Per	Come
		zione Centro identità IAM nella Guida di riferimento per SDK e strumenti AWS.
IAM	Utilizza credenziali temporane e per firmare richieste programmatiche alla AWS CLI, agli SDK AWS o alle API AWS.	Segui le istruzioni in Utilizzo di credenziali temporanee con le risorse AWS nella Guida per l'utente IAM.
IAM	(Non consigliato) Utilizza credenziali a lungo termine per firmare richieste programmatiche alla AWS CLI, agli SDK AWS o alle API AWS.	Segui le istruzioni per l'interfa ccia che desideri utilizzare. • Per la AWS CLI, consulta la pagina Autenticazione tramite credenziali utente IAM nella Guida per l'utente dell'AWS Command Line Interface. • Per gli SDK e gli strumenti AWS, consulta la pagina Autenticazione con credenziali a lungo termine nella Guida di riferimento per SDK e strumenti AWS. • Per le API AWS, consulta la pagina Gestione delle chiavi di accesso per utenti IAM nella Guida per l'utente IAM.

Fase 3: ottieni un codice di esempio pronto per l'uso

Questa guida include esempi che utilizzano l'AWS SDK per Java. Per eseguire il codice di esempio, segui le istruzioni di configurazione in Guida introduttiva a AWS SDK per Java 2.0.

È possibile sviluppare AWS applicazioni in altri linguaggi di programmazione, come GoJavaScript, Python e Ruby. Per ulteriori informazioni, consulta Strumenti per lo sviluppo e la gestione di applicazioni su AWS.

> **Note**
>
> Puoi esplorare Amazon SQS senza scrivere codice con strumenti come AWS Command Line Interface (AWS CLI) o Windows. PowerShell Puoi trovare esempi AWS CLI nella sezione Amazon SQS della Documentazione di riferimento ai comandi AWS CLI. Puoi trovare PowerShell esempi di Windows nella sezione Amazon Simple Queue Service del AWS Tools for PowerShellCmdlet Reference.

Passaggi successivi

Ora sei pronto per iniziare a gestire le code e i messaggi di Amazon SQS utilizzando AWS Management Console.

Nozioni di base su Amazon SQS

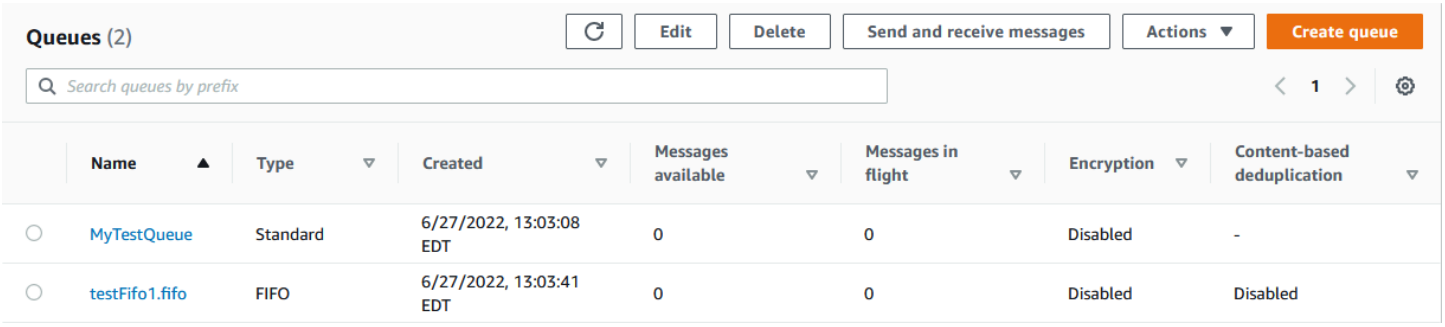
In questa sezione, imparerai come creare code standard o FIFO utilizzando la console Amazon SQS.

Prerequisiti

Prima di iniziare, completa i passaggi descritti in Configurazione di Amazon SQS.

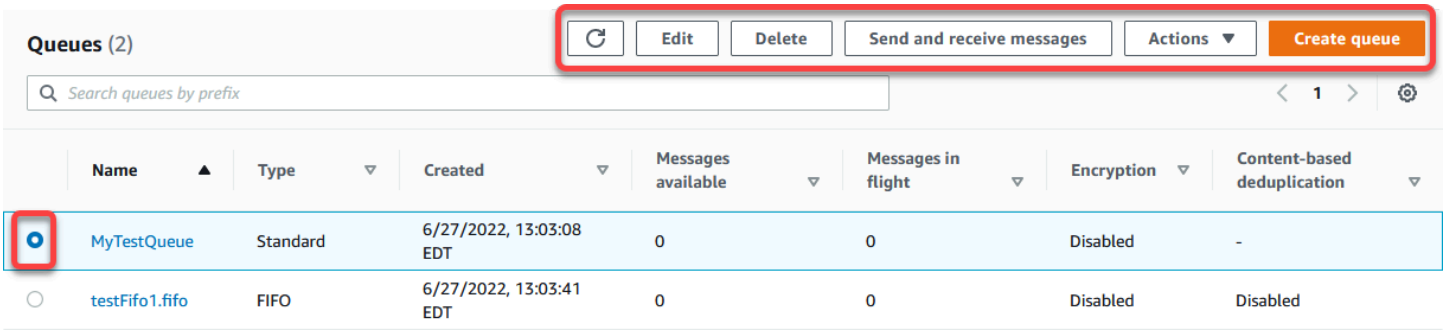
Comprendere le nozioni di base della console Amazon SQS

Quando apri la console, scegli Code dal pannello di navigazione per visualizzare la pagina Code. La pagina Code fornisce informazioni su tutte le code nell'area attiva.

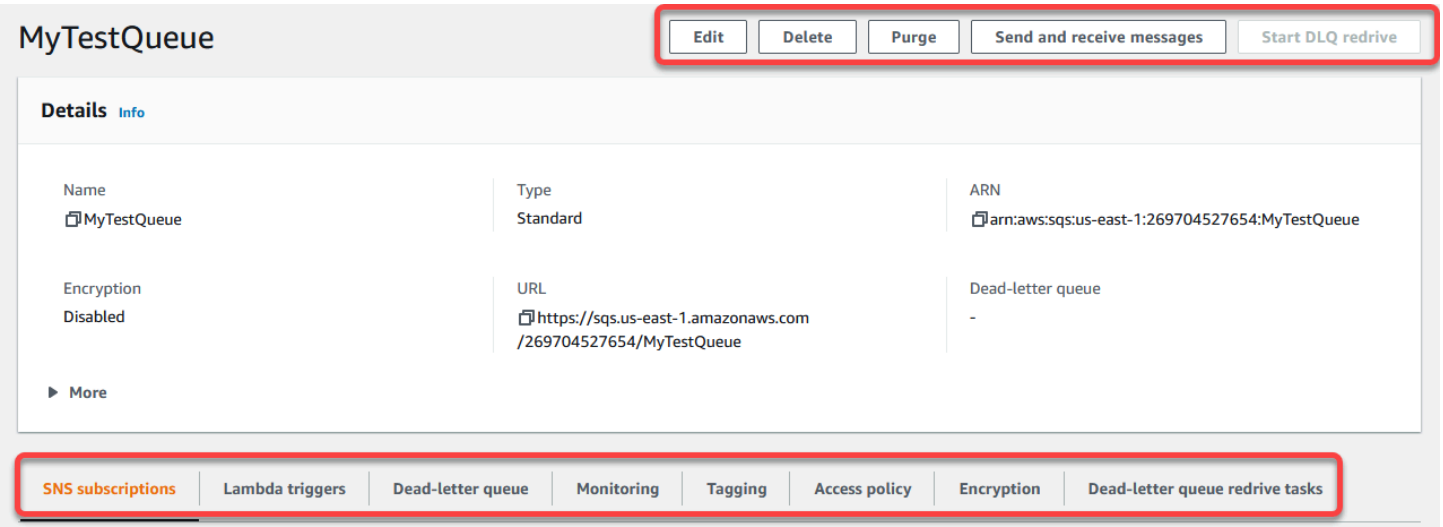


La voce relativa a ciascuna coda mostra il tipo di coda e altre informazioni sulla coda. La colonna Tipo consente di distinguere a colpo d'occhio le code standard dalle code First-In-First Out (FIFO).

Dalla pagina Code, ci sono due modi per eseguire azioni su una coda. È possibile scegliere l'opzione accanto al nome della coda e quindi scegliere l'azione che si desidera eseguire sulla coda.



Puoi anche scegliere il nome della coda, che apre la pagina Dettagli della coda. La pagina Dettagli include le stesse azioni della pagina Code. Inoltre, è possibile scegliere una delle schede sotto la sezione Dettagli per visualizzare ulteriori dettagli e azioni di configurazione.



Tipi di code Amazon SQS

Amazon SQS supporta due tipi di code: code standard e code FIFO. Utilizza le informazioni della tabella seguente per scegliere la coda giusta per la tua situazione. Per ulteriori informazioni sulle code di Amazon SQS, consulta Nozioni di base sulle code standard di Amazon SQS e Nozioni di base sulle code FIFO di Amazon SQS.

Code standard	Code FIFO
Velocità di trasmissione effettiva illimitata: le code standard supportano un numero quasi illimitato di chiamate API al secondo, per azione API (`SendMessage` , `ReceiveMessage` o `DeleteMessage`). Consegna "At-Least-Once": un messaggio viene consegnato almeno una volta, ma di tanto in tanto viene consegnata più di una copia di un messaggio. Miglior ordine possibile: di tanto in tanto i messaggi potrebbero essere consegnati in un ordine diverso da quello di invio.	Velocità di trasmissione effettiva elevata: se utilizzi il batch, le code FIFO supportano fino a 3.000 messaggi al secondo per metodo API (`SendMessageBatch` , `ReceiveMessage` o `DeleteMessageBatch`). I 3.000 messaggi rappresentano 300 chiamate API, ognuna con un batch di 10 messaggi. Per richiedere un incremento della quota, invia una richiesta di supporto. Senza batch, le code FIFO supportano fino a 300 chiamate API al secondo, per metodo API (`SendMessage` , `ReceiveMessage` o `DeleteMessage`). Elaborazione "exactly-once": un messaggio viene consegnato una volta e rimane disponibile fino a quando un cliente non lo elabora e lo

Code standard	Code FIFO
	cancella. I duplicati non vengono introdotti nella coda. Consegna First-In-First-Out: viene mantenuto l'ordine esatto di invio e ricezione dei messaggi.
	
Invio di dati tra applicazioni quando il throughput è importante, ad esempio: • Disassociare le richieste utente live da lavori intensivi: consente agli utenti di caricare risorse multimediali mentre vengono ridimensionate o codificate. • Assegnare le attività a nodi lavoratore multipli: elaborare un elevato numero di richieste di convalida di carta di credito. • Messaggi in batch per elaborazione futura: programmare voci multiple da aggiungere a un database.	Invio di dati tra applicazioni quando l'ordine degli eventi è importante, ad esempio: • Accertati che i comandi immessi dall'utente vengano eseguiti nel giusto ordine. • Visualizzare il prezzo corretto del prodotto inviando le modifiche di prezzo nel giusto ordine. • Impedire a uno studente di iscriversi a un corso prima di registrarsi per un account.

Creazione di una coda Amazon SQS standard e invio di un messaggio

Ecco come creare una coda standard per Amazon SQS.

Creazione di una coda (console)

È possibile utilizzare la console Amazon SQS per creare code standard. La console fornisce valori predefiniti per tutte le impostazioni ad eccezione del nome della coda.

> **Important**
>
> Il 17 agosto 2022, la crittografia lato server predefinita (SSE) è stata applicata a tutte le code Amazon SQS.
> Non aggiungere informazioni di identificazione personale (PII) o altre informazioni riservate o sensibili nei nomi delle code. I nomi delle code sono accessibili a molti Amazon Web Services, inclusi fatturazione e CloudWatch log. I nomi delle code non sono destinati a essere utilizzati per dati privati o sensibili.

Creazione di una coda standard Amazon SQS

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Scegliere Crea coda.
3. Per Tipo, il tipo di coda Standard è impostato di default.

   > **Note**
   >
   > Il tipo di coda non può essere modificato dopo la creazione della coda.

4. Inserisci un Nome per la coda.
5. (Facoltativo) La console imposta i valori predefiniti per i parametri di configurazione della coda. In Configurazione, è possibile impostare nuovi valori per i seguenti parametri:

   a. Per il timeout di visibilità, inserisci la durata e le unità. L'intervallo è compreso tra 0 secondi e 12 ore. Il valore di predefinito è 30 secondi.

   b. Per Periodo di conservazione dei messaggi, inserisci la durata e le unità. L'intervallo valido è compreso tra 1 minuto e 14 giorni. Il valore predefinito è 4 giorni.

   c. Per Ritardo di consegna, inserisci la durata e le unità. L'intervallo è compreso tra 0 secondi e 15 minuti. Il valore predefinito è 0 secondi.

   d. Per Dimensione massima del messaggio, inserisci un valore. L'intervallo è compreso tra 1 e 256 KB. Il valore predefinito è 256 KB.

   e. Per Tempo di attesa per la ricezione del messaggio, inserisci un valore. L'intervallo è tra 0 e 20 secondi. Il valore predefinito è 0 secondi, che imposta uno short polling. Qualsiasi valore diverso da zero imposta un long polling.

6. (Facoltativo) Definire una policy di accesso. La policy di accesso definisce gli account, gli utenti e i ruoli che possono accedere alla coda. La policy di accesso definisce anche le operazioni (ad esempio `SendMessage`, `ReceiveMessage` o`DeleteMessage`) a cui gli utenti possono accedere. La policy predefinita consente solo al proprietario della coda di inviare e ricevere messaggi.

   Per definire la policy di accesso, effettua una delle seguenti operazioni:

   - Scegli Basic per configurare chi può inviare messaggi alla coda e chi può ricevere messaggi dalla coda. La console crea la policy in base alle tue scelte e visualizza la policy di accesso risultante nel pannello JSON di sola lettura.
   - Scegli Avanzato per modificare direttamente la policy di accesso JSON. Ciò consente di specificare un set personalizzato di azioni che ogni principale (account, utente o ruolo) può eseguire.
7. Per la policy Redrive allow, scegli Abilitata. Seleziona una delle seguenti opzioni: Consenti tutto, Per coda o Nega tutto. Quando scegli Per coda, specifica un elenco di un massimo di 10 code di origine in base al nome della risorsa Amazon (ARN).
8. Amazon SQS fornisce la crittografia lato server gestita per impostazione predefinita. Per scegliere un tipo di chiave di crittografia o per disabilitare la crittografia lato server gestita da Amazon SQS, espandi Crittografia. Per ulteriori informazioni sui tipi di chiavi di crittografia, consulta Configurazione della crittografia lato server (SSE) per una coda tramite chiavi di crittografia gestite da SQS (console) e Configurazione della crittografia lato server (SSE) per una coda (console).

   > **Note**
   >
   > Con SSE abilitato, le richieste anonime `SendMessage` e `ReceiveMessage` alla coda crittografata verranno rifiutate. Le best practice di sicurezza di Amazon SQS consigliano di non utilizzare richieste anonime. Se desideri inviare richieste anonime a una coda Amazon SQS, assicurati di disabilitare SSE.
9. (Facoltativo) Per configurare una coda DLQ per la ricezione di messaggi non recapitabili, espandere Coda DLQ.
10. (Facoltativo) Per aggiungere tag alla coda, espandi Tag.
11. Scegliere Crea coda. Amazon SQS crea la coda e visualizza la pagina dei dettagli della coda.

Amazon SQS diffonde le informazioni sulla nuova coda in tutto il sistema. Poiché Amazon SQS è un sistema distribuito, potrebbe verificarsi un leggero ritardo prima che la console visualizzi la coda nella pagina Code.

Invio di un messaggio

Dopo aver creato la coda, puoi inviarle un messaggio.

1. Nel riquadro di navigazione sinistro scegliere Code. Nell'elenco delle code, seleziona la coda appena creata.
2. Scegliere Operazioni, quindi Invia e ricevi messaggi.

   La console visualizza la pagina Invia e ricevi messaggi.
3. In Corpo del messaggio, inserisci il testo del messaggio.
4. Per una coda standard, puoi inserire un valore per Ritardo di consegna e scegliere le unità. Ad esempio, inserisci `60` e scegli i secondi. Per ulteriori informazioni, consulta Timer di messaggi Amazon SQS.
5. Scegliere Invia messaggio.

   Quando il messaggio viene inviato, la console ne visualizza la conferma. Scegli Visualizza dettagli per visualizzare le informazioni sul messaggio inviato.

Creazione di una coda FIFO Amazon SQS e invio di un messaggio

Ecco come creare una coda FIFO per Amazon SQS.

Crea una coda

È possibile utilizzare la console Amazon SQS per creare le code FIFO. La console fornisce valori predefiniti per tutte le impostazioni ad eccezione del nome della coda.

> **Important**
>
> Il 17 agosto 2022, la crittografia lato server predefinita (SSE) è stata applicata a tutte le code Amazon SQS.
> Non aggiungere informazioni di identificazione personale (PII) o altre informazioni riservate o sensibili nei nomi delle code. I nomi delle code sono accessibili a molti Amazon Web

Services, inclusi fatturazione e CloudWatch log. I nomi delle code non sono destinati a essere utilizzati per dati privati o sensibili.

Creazione di una coda FIFO Amazon SQS

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Scegliere Crea coda.
3. Per Tipo, il tipo di coda Standard è impostato di default. Per creare una coda FIFO, scegliere FIFO.

   > **Note**
   >
   > Il tipo di coda non può essere modificato dopo la creazione della coda.

4. Inserisci un Nome per la coda.

   Il nome di una coda FIFO deve terminare con il suffisso `.fifo`. Il suffisso viene conteggiato ai fini della quota di 80 caratteri dei nomi della coda. Per determinare se una coda è FIFO, puoi verificare se il nome della coda termina con il suffisso.

5. (Facoltativo) La console imposta i valori predefiniti per i parametri di configurazione della coda. In Configurazione, è possibile impostare nuovi valori per i seguenti parametri:

   a. Per il timeout di visibilità, inserisci la durata e le unità. L'intervallo è compreso tra 0 secondi e 12 ore. Il valore di predefinito è 30 secondi.

   b. Per Periodo di conservazione dei messaggi, inserisci la durata e le unità. L'intervallo valido è compreso tra 1 minuto e 14 giorni. Il valore predefinito è 4 giorni.

   c. Per Ritardo di consegna, inserisci la durata e le unità. L'intervallo è compreso tra 0 secondi e 15 minuti. Il valore predefinito è 0 secondi.

   d. Per Dimensione massima del messaggio, inserisci un valore. L'intervallo è compreso tra 1 e 256 KB. Il valore predefinito è 256 KB.

   e. Per Tempo di attesa per la ricezione del messaggio, inserisci un valore. L'intervallo è tra 0 e 20 secondi. Il valore predefinito è 0 secondi, che imposta uno short polling. Qualsiasi valore diverso da zero imposta un long polling.

   f. Per una coda FIFO, scegli Deduplicazione basata sul contenuto per abilitare la deduplicazione basata sul contenuto. L'impostazione predefinita è disabilitata.

g. (Facoltativo) Affinché una coda FIFO consenta una velocità di trasmissione effettiva più elevata per l'invio e la ricezione di messaggi in coda, seleziona Abilita FIFO ad alta velocità di trasmissione effettiva.

   La scelta di questa opzione modifica le opzioni correlate (ambito di deduplicazione e limite di velocità di trasmissione effettiva FIFO) con le impostazioni richieste per abilitare una velocità di trasmissione effettiva elevata per le code FIFO. Se si modifica una delle impostazioni necessarie per utilizzare FIFO ad alta velocità di trasmissione effettiva, si applica la velocità di trasmissione effettiva normale per la coda e la deduplicazione si verifica come specificato. Per ulteriori informazioni, consultare Velocità di trasmissione effettiva elevata per le code FIFO e Quote correlate ai messaggi.

6. (Facoltativo) Definire una policy di accesso. La policy di accesso definisce gli account, gli utenti e i ruoli che possono accedere alla coda. La policy di accesso definisce anche le operazioni (ad esempio `SendMessage`, `ReceiveMessage` o`DeleteMessage`) a cui gli utenti possono accedere. La policy predefinita consente solo al proprietario della coda di inviare e ricevere messaggi.

   Per definire la policy di accesso, effettua una delle seguenti operazioni:

   - Scegli Basic per configurare chi può inviare messaggi alla coda e chi può ricevere messaggi dalla coda. La console crea la policy in base alle tue scelte e visualizza la policy di accesso risultante nel pannello JSON di sola lettura.
   - Scegli Avanzato per modificare direttamente la policy di accesso JSON. Ciò consente di specificare un set personalizzato di azioni che ogni principale (account, utente o ruolo) può eseguire.

7. Per la policy Redrive allow, scegli Abilitata. Seleziona una delle seguenti opzioni: Consenti tutto, Per coda o Nega tutto. Quando scegli Per coda, specifica un elenco di un massimo di 10 code di origine in base al nome della risorsa Amazon (ARN).

8. Amazon SQS fornisce la crittografia lato server gestita per impostazione predefinita. Per scegliere un tipo di chiave di crittografia o per disabilitare la crittografia lato server gestita da Amazon SQS, espandi Crittografia. Per ulteriori informazioni sui tipi di chiavi di crittografia, consulta Configurazione della crittografia lato server (SSE) per una coda tramite chiavi di crittografia gestite da SQS (console) e Configurazione della crittografia lato server (SSE) per una coda (console).

> **Note**
>
> Con SSE abilitato, le richieste anonime `SendMessage` e `ReceiveMessage` alla coda crittografata verranno rifiutate. Le best practice di sicurezza di Amazon SQS consigliano di non utilizzare richieste anonime. Se desideri inviare richieste anonime a una coda Amazon SQS, assicurati di disabilitare SSE.

9. (Facoltativo) Per configurare una coda DLQ per la ricezione di messaggi non recapitabili, espandere Coda DLQ.
10. (Facoltativo) Per aggiungere tag alla coda, espandi Tag.
11. Scegliere Crea coda. Amazon SQS crea la coda e visualizza la pagina dei dettagli della coda.

Amazon SQS diffonde le informazioni sulla nuova coda in tutto il sistema. Poiché Amazon SQS è un sistema distribuito, potrebbe verificarsi un leggero ritardo prima che la console visualizzi la coda nella pagina Code.

Dopo aver creato una coda, puoi inviarle messaggi e riceverli ed eliminarli. È inoltre possibile modificare qualsiasi impostazione di configurazione della coda ad eccezione del tipo di coda.

Invio di un messaggio

Dopo aver creato la coda, puoi inviarle un messaggio.

1. Nel riquadro di navigazione sinistro scegliere Code. Nell'elenco delle code, seleziona la coda appena creata.
2. Scegliere Operazioni, quindi Invia e ricevi messaggi.

   La console visualizza la pagina Invia e ricevi messaggi.
3. In Corpo del messaggio, inserisci il testo del messaggio.
4. Per una coda First-Out (FIFO), inserisci l'ID del gruppo di messaggi. Per ulteriori informazioni, consulta Logica di distribuzione FIFO.
5. (Facoltativo) Per una coda FIFO, puoi inserire un ID di deduplicazione dei messaggi. Se hai abilitato la deduplicazione basata sul contenuto per la coda, l'ID di deduplicazione dei messaggi non è richiesto. Per ulteriori informazioni, consulta Logica di distribuzione FIFO.
6. Le code FIFO non supportano i timer sui singoli messaggi. Per ulteriori informazioni, consulta Timer di messaggi Amazon SQS.

7. Scegliere Invia messaggio.

   Quando il messaggio viene inviato, la console ne visualizza la conferma. Scegli Visualizza dettagli per visualizzare le informazioni sul messaggio inviato.

Gestione di una coda Amazon SQS

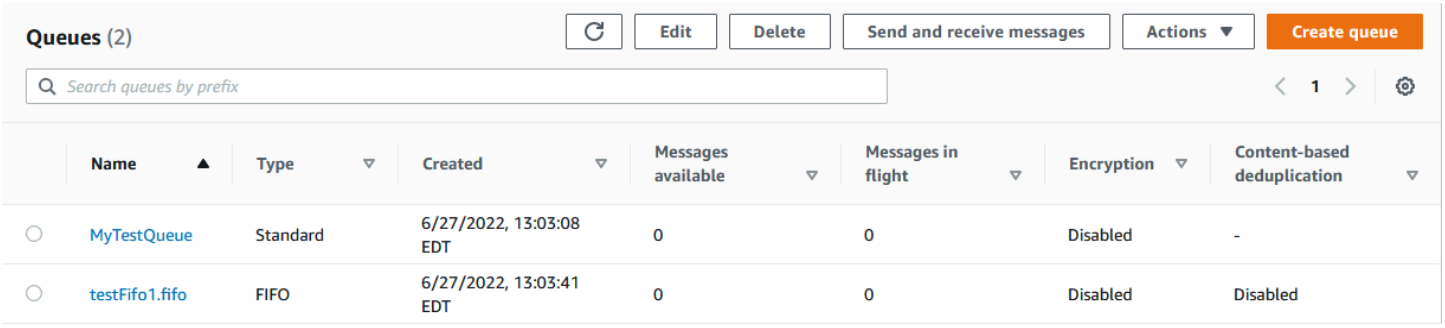
Questa sezione consente di familiarizzare con Amazon SQS mostrando come gestire code e messaggi utilizzando la console di Amazon SQS.

Prerequisiti

Prima di iniziare, completa i passaggi descritti in Configurazione di Amazon SQS.

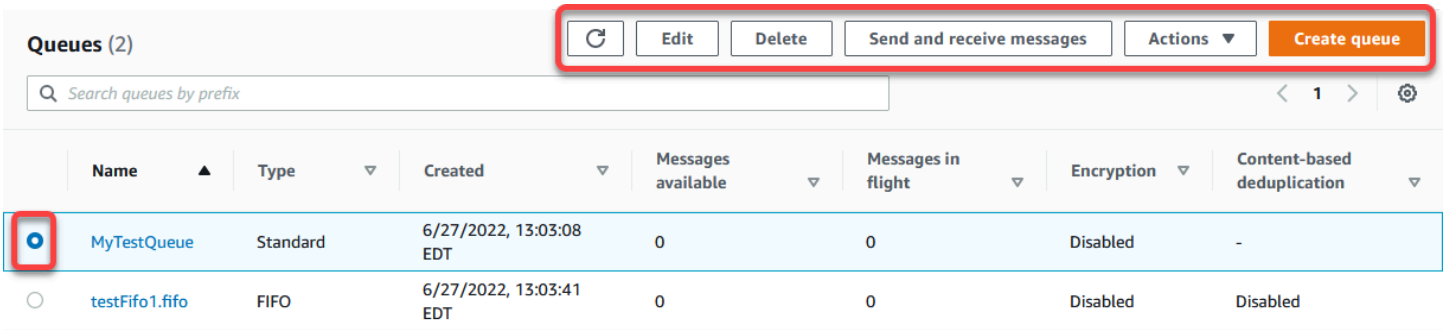
Comprendere le nozioni di base della console Amazon SQS

Quando apri la console, scegli Code dal pannello di navigazione per visualizzare la pagina Code. La pagina Code fornisce informazioni su tutte le code nell'area attiva.

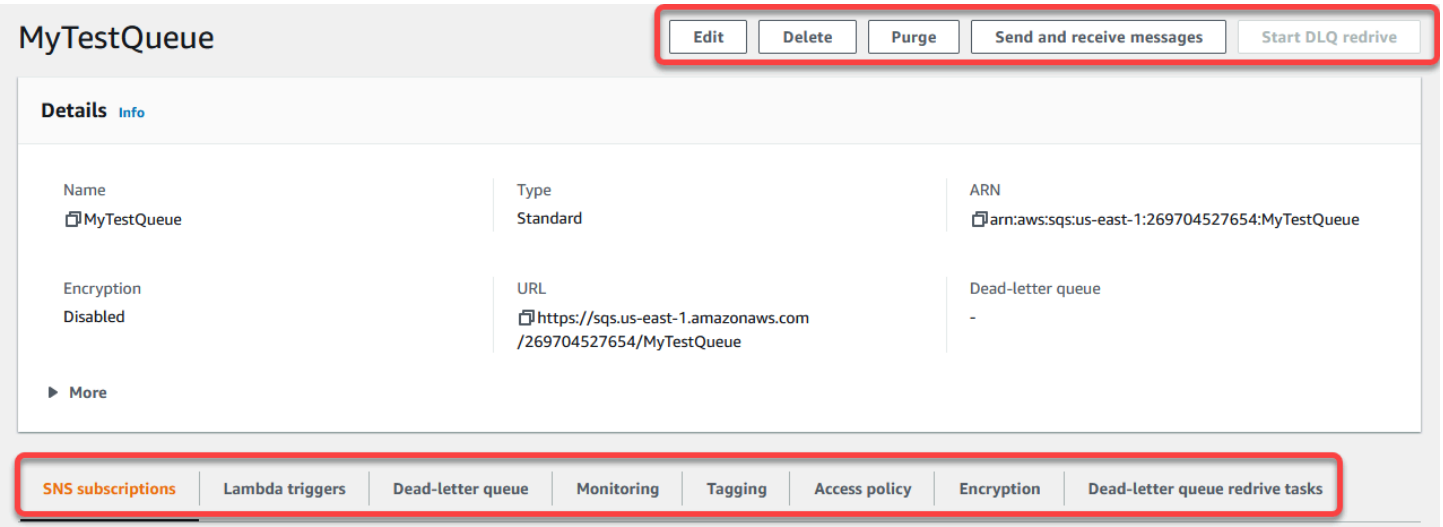


La voce relativa a ciascuna coda mostra il tipo di coda e altre informazioni sulla coda. La colonna Tipo consente di distinguere a colpo d'occhio le code standard dalle code First-In-First Out (FIFO).

Dalla pagina Code, ci sono due modi per eseguire azioni su una coda. È possibile scegliere l'opzione accanto al nome della coda e quindi scegliere l'azione che si desidera eseguire sulla coda.



Puoi anche scegliere il nome della coda, che apre la pagina Dettagli della coda. La pagina Dettagli include le stesse azioni della pagina Code. Inoltre, è possibile scegliere una delle schede sotto la sezione Dettagli per visualizzare ulteriori dettagli e azioni di configurazione.



Modificare una coda (console)

Puoi utilizzare la console Amazon SQS per modificare qualsiasi parametro di configurazione della coda (eccetto il tipo di coda) e aggiungere o rimuovere funzionalità di coda.

Per modificare una coda Amazon SQS (console)

1. Nella console Amazon SQS, apri la pagina delle code.
2. Seleziona un argomento, quindi scegli Modifica.
3. (Facoltativo) In Configurazione, aggiorna i parametri di configurazione della coda.
4. (Facoltativo) Per aggiornare la policy di accesso, in Policy di accesso, modifica la policy JSON.
5. (Facoltativo) Per aggiornare una policy redrive allow di una coda DLQ, espandi la policy redrive allow.
6. (Facoltativo) Per aggiornare o rimuovere la crittografia, espandi Crittografia.
7. (Facoltativo) Per aggiungere, aggiornare o rimuovere una coda DLQ (che consente di ricevere messaggi non recapitabili), espandi la coda DLQ.
8. (Facoltativo) Per aggiungere, aggiornare o rimuovere i tag per la coda, espandi Tag.
9. Selezionare Salva.

   La console visualizza la pagina Dettagli per la coda.

Ricevere ed eliminare un messaggio (console)

Dopo aver inviato i messaggi a una coda, puoi riceverli ed eliminarli. Quando richiedi messaggi da una coda, non puoi specificare quali messaggi recuperare. Puoi invece indicare il numero massimo di messaggi (fino a 10) da recuperare.

> **Note**
>
> Poiché Amazon SQS è un sistema distribuito, una coda con pochissimi messaggi potrebbe mostrare una risposta vuota a una richiesta di ricezione. In questo caso, esegui nuovamente la richiesta per ricevere il messaggio. A seconda delle esigenze dell'applicazione, potrebbe essere necessario utilizzare polling brevi o lunghi per ricevere messaggi.

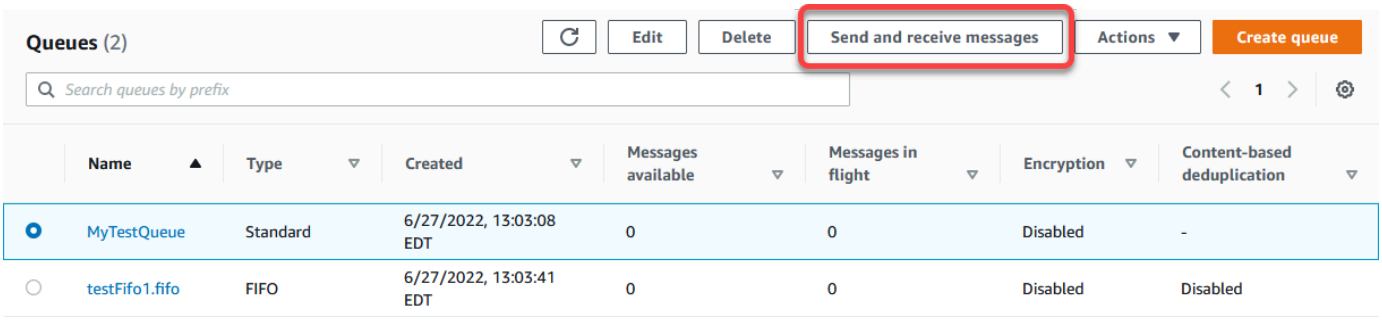
Amazon SQS non elimina automaticamente un messaggio dopo la ricezione al tuo posto se non dovessi riceverlo (ad esempio, i consumatori possono avere problemi di connessione scarsa o assente). Per eliminare un messaggio, devi inviare una richiesta separata che attesta che hai ricevuto ed elaborato il messaggio in modo corretto. Si noti che è necessario ricevere un messaggio prima di poterlo eliminare.

> **Note**
>
> Dopo aver ricevuto messaggi dalla console Amazon SQS, la console rende immediatamente i messaggi visibili, in modo che possano essere nuovamente ricevuti.

Per ulteriori informazioni sulle opzioni API per la ricezione e l'eliminazione dei messaggi, consulta la Amazon SQS API Reference Guide.

Ricevere ed eliminare un messaggio (console)

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code).
3. Nella pagina Code, scegliere una coda.
4. Scegli Invia e ricevi messaggi.



La console visualizza la pagina Invia e ricevi messaggi.

5. Scegli Sondaggio per i messaggi.

   Amazon SQS inizia a verificare la presenza di messaggi in coda. La barra di avanzamento sul lato destro della sezione Ricezione messaggi mostra la durata del polling.

   La sezione Messaggi mostra un elenco dei messaggi ricevuti. Per ogni messaggio, l'elenco mostra l'ID del messaggio, la data di invio, la dimensione e il numero di ricezione.

6. Per eliminare i messaggi, scegliere i messaggi da eliminare e scegliere Elimina.
7. Nella finestra di dialogo Elimina messaggi, scegli Elimina.

Verifica che una coda sia vuota

Nella maggior parte dei casi, è possibile utilizzare un long polling per determinare se una coda è vuota. In rari casi, potresti ricevere risposte vuote anche quando una coda contiene ancora messaggi, specialmente se hai specificato un valore basso per il tempo di attesa per la ricezione dei messaggi quando hai creato la coda. Questa sezione spiega come confermare che una coda sia vuota.

Verifica che una coda sia vuota (console)

1. Impedisci a tutti i produttori di inviare messaggi.
2. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
3. Nel riquadro di navigazione, scegliere Code (Code).
4. Nella pagina Code, scegliere una coda.
5. Scegliere la scheda Monitoring (Monitoraggio).
6. In alto a destra nelle dashboard di monitoraggio, scegli la freccia rivolta verso il basso accanto al simbolo Aggiorna. Dal menu a discesa scegli Aggiornamento automatico. Lascia l'intervallo di aggiornamento a 1 minuto.

7. Osserva le seguenti dashboard:

   - Numero approssimativo di messaggi ritardati
   - Numero approssimativo di messaggi non visibili
   - Numero approssimativo di messaggi visibili

   Quando tutti mostrano valori `0` per diversi minuti, la coda è vuota.

Per confermare che una coda è vuota (AWS CLI, AWS API)

1. Impedisci a tutti i produttori di inviare messaggi.
2. Eseguire ripetutamente uno dei seguenti comandi:

   - AWS CLI: `get-queue-attributes`
   - AWS API: `GetQueueAttributes`
3. Osserva le metriche per gli attributi seguenti:

   - `ApproximateNumberOfMessagesDelayed`
   - `ApproximateNumberOfMessagesNotVisible`
   - `ApproximateNumberOfMessagesVisible`

   Quando tutti mostrano valori `0` per diversi minuti, la coda è vuota.

Se ti affidi ai CloudWatch parametri di Amazon, assicurati di visualizzare più punti dati zero consecutivi prima di considerare la coda vuota. Per ulteriori informazioni sui CloudWatch parametri, consulta. CloudWatch Metriche disponibili per Amazon SQS

Elimina una coda

Se non usi più una coda Amazon SQS e non prevedi di utilizzarla in un prossimo futuro, ti consigliamo di eliminarla.

> **Tip**
>
> Se vuoi verificare che una coda sia vuota prima di eliminarla, consulta Verifica che una coda sia vuota.

È possibile eliminare una coda anche quando non è vuota. Se desideri eliminare i messaggi in una coda, ma non la coda, è possibile svuotare la coda.

Per eliminare una coda (console)

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code).
3. Nella pagina Code, scegliere la coda da eliminare.
4. Scegli Elimina.
5. Nella finestra di dialogo Elimina coda, conferma l'eliminazione inserendo **`delete`**.
6. Scegli Elimina.

Per eliminare una coda (API) AWS CLIAWS

Per eliminare una coda, puoi utilizzare uno dei seguenti comandi:

- AWS CLI: `aws sqs delete-queue`
- AWS API: `DeleteQueue`

Eliminazione dei messaggi da una coda Amazon SQS (console)

Se non vuoi eliminare una coda Amazon SQS, ma devi eliminare tutti i messaggi dalla stessa, puoi rimuovere la coda. Il processo di eliminazione dei messaggi richiede fino a 60 secondi. Ti consigliamo di attendere 60 secondi indipendentemente dalle dimensioni della coda.

> **Important**
>
> Quando elimini una coda, non puoi recuperare i messaggi eliminati.

Eliminare una coda (console)

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code).
3. Nella pagina Code, scegliere la coda da eliminare.
4. In Operazioni, scegliere Eliminare.
5. Nella finestra di dialogo Elimina coda, conferma l'eliminazione inserendo **`purge`** e scegliendo Elimina.

   Tutti i messaggi vengono rimossi dalla coda. Nella console viene visualizzato un banner di conferma.

Attività comuni per iniziare a usare Amazon SQS

Ora che hai creato una coda e appreso come inviare, ricevere ed eliminare messaggi e come eliminare una coda, puoi provare quanto segue:

- Per attivare una funzione Lambda, vedere Configurazione di una coda per attivare una funzione AWS Lambda (console).
- Scopri come configurare le code, tra cui SSE e altre funzionalità.
- Scopri come inviare un messaggio con attributi.
- Scopri come inviare un messaggio da un VPC.
- Per scoprire le funzionalità e l'architettura di Amazon SQS, consulta Tipi di code Amazon SQS e Architettura Amazon SQS di base.
- Per scoprire le linee guida e gli avvertimenti che consentono di sfruttare al massimo Amazon SQS, consulta Best practice per Amazon SQS.
- Esplora gli esempi di Amazon SQS per uno degli AWS SDK, come la Developer Guide. AWS SDK for Java 2.x
- Per ulteriori informazioni sui AWS CLI comandi di Amazon SQS, consulta il AWS CLI Command Reference.
- Per ulteriori informazioni sulle operazioni di Amazon SQS, consulta la Guida di riferimento per le API di Amazon Simple Queue Service.
- Scopri come interagire con Amazon SQS in modo programmatico: leggi Lavorare con le API ed esplora Codice e librerie di esempio e centri sviluppatori:

- Java
- JavaScript
- PHP
- Python
- Ruby
- Windows e .NET

- Ulteriori informazioni su costi e risorse da tenere sotto controllo nella sezione Automazione e risoluzione dei problemi delle code di Amazon SQS.
- Ulteriori informazioni sulla protezione e l'accesso ai dati nella sezione Sicurezza.
- Scopri di più sui flussi di lavoro e sui processi di Amazon SQS:

Nozioni di base sulle code standard di Amazon SQS

Amazon SQS offre standard come tipo di coda predefinito. Le code standard supportano un numero quasi illimitato di chiamate API al secondo, per azione API (`SendMessage`, `ReceiveMessage` o `DeleteMessage`). Le code standard supportano il recapito dei at-least-once messaggi. Tuttavia, occasionalmente (a causa dell'architettura altamente distribuita che consente un throughput quasi illimitato), è possibile che più di una copia di un messaggio venga consegnata fuori ordine. Le code standard forniscono l'ordine migliore, che garantisce che i messaggi siano generalmente consegnati nello stesso ordine in cui vengono inviati.

Amazon SQS archivia in modo ridondante un messaggio in più di una zona di disponibilità (AZ) prima che venga riconosciuto un `SendMessage`. Poiché le copie dei messaggi sono archiviate in più AZ, nessun errore in un singolo computer, rete o AZ può rendere i messaggi inaccessibili.

Per informazioni su come creare e configurare code utilizzando la console Amazon SQS, consulta Creazione di una coda (console). Per esempi in Java, consulta Esempi di SDK Java di Amazon SQS.

Puoi utilizzare code di messaggi standard in molti scenari, purché l'applicazione sia in grado di elaborare i messaggi che arrivano più di una volta e nell'ordine sbagliato. Ad esempio:

- Disassociare le richieste utente live da lavori intensivi: consente agli utenti di caricare risorse multimediali mentre vengono ridimensionate o codificate.
- Assegnare le attività a nodi lavoratore multipli: elaborare un elevato numero di richieste di convalida di carta di credito.
- Messaggi in batch per elaborazione futura: programmare voci multiple da aggiungere a un database.

Per le quote relative alle code standard, consulta Quote.

Per le best practice relative all'utilizzo di code standard, consulta Consigli per le code FIFO e standard di Amazon SQS.

Ordine dei messaggi

Una coda standard si sforza di preservare l'ordine dei messaggi, ma più di una copia di un messaggio potrebbe essere distribuita fuori ordine. Se il sistema richiede che l'ordine venga conservato, è

consigliato l'uso di una coda FIFO (First-In-First-Out) o l'aggiunta di informazioni di sequenziamento in ogni messaggio in modo da riordinare i messaggi ricevuti.

Una consegna t-least-once

Amazon SQS archivia copie dei messaggi su più server per ridondanza e disponibilità elevata. In rare occasioni, uno dei server che memorizza una copia di un messaggio potrebbe non essere disponibile quando ricevi o elimini un messaggio.

In tal caso, la copia del messaggio non viene eliminata sul server non disponibile, e potresti ricevere di nuovo la copia del messaggio quando ricevi i messaggi. Progetta le tue applicazioni affinché siano idempotent (non devono essere condizionate negativamente quando elaborano lo stesso messaggio più di una volta).

Identificatori di code e messaggi Amazon SQS

Questa sezione descrive gli identificatori delle code standard e FIFO. Questi identificatori possono aiutarti a trovare e modificare code e messaggi specifici.

Argomenti

- Identificatori per le code standard di Amazon SQS

Identificatori per le code standard di Amazon SQS

Per ulteriori informazioni, consulta i seguenti argomenti nella Documentazione di riferimento delle API di Amazon Simple Queue Service.

Nome e URL della coda

Quando crei una nuova coda, è necessario specificare un nome coda univoco per il tuo account e la tua regione AWS. Amazon SQS assegna a ogni coda che crei un identificatore chiamato URL coda che include il nome della coda e altri componenti Amazon SQS. Se desideri eseguire un'operazione su una coda, devi fornire il relativo URL coda.

Di seguito è riportato l'URL coda per una coda denominata `MyQueue`, di proprietà di un utente con il numero di account AWS `123456789012`.

```
https://sqs.us-east-2.amazonaws.com/123456789012/MyQueue
```

È possibile recuperare l'URL di una coda a livello di codice elencando le code e analizzando la stringa che segue il numero di account. Per ulteriori informazioni, consulta `ListQueues`.

ID messaggio

Ogni messaggio riceve un ID messaggio assegnato dal sistema che Amazon SQS restituisce nella risposta `SendMessage`. Questo identificatore è utile per l'identificazione di messaggi. La durata massima di un ID messaggio è di 100 caratteri.

Handle di ricezione

Ogni volta che ricevi un messaggio da una coda, ricevi un handle di ricezione per tale messaggio. Questo handle è associato all'operazione di ricezione del messaggio, non al messaggio stesso. Per eliminare il messaggio o per modificarne la visibilità, devi fornire l'handle di ricezione (non l'ID messaggio). Pertanto, devi sempre ricevere un messaggio prima di poterlo eliminare (non puoi inserire un messaggio nella coda e poi richiamarlo). La durata massima di un handle di ricezione è di 1024 caratteri.

> **Important**
>
> Se ricevi un messaggio più di una volta, ogni volta che lo ricevi ottieni un diverso handle di ricezione. Devi fornire l'handle di ricezione ricevuto più di recente quando richiedi di eliminare il messaggio (in caso contrario, il messaggio potrebbe non essere eliminato).

Di seguito è riportato un esempio di handle di ricezione (suddiviso su tre linee).

```
MbZj6wDWli+JvwwJaBV+3dcjk2YW2vA3+STFFljTM8tJJg6HRG6PYSasuWXPJB+Cw
Lj1FjgXUv1uSj1gUPAWV66FU/WeR4mq2OKpEGYWbnLmpRCJVAyeMjeU5ZBdtcQ+QE
auMZc8ZRv37sIW2iJKq3M9MFx1YvV11A2x/KSbkJ0=
```

Quote

La tabella seguente elenca le quote correlate alle code standard.

Quota	Descrizione
Coda di ritardo	Il ritardo predefinito (minimo) per una coda è 0 secondi. Il valore massimo è 15 minuti.

Quota	Descrizione
Code elencate	1.000 code per ogni richiesta `ListQueues` .
Tempo di attesa del long polling	Il tempo massimo di attesa per il long polling è di 20 secondi.
Messaggi per coda (backlog)	Il numero di messaggi che una coda Amazon SQS può archiviare è illimitato.
Messaggi per coda (in transito)	Per la maggior parte delle code standard (a seconda del traffico in coda e del backlog di messaggi), possono esserci un massimo di circa 120.000 messaggi in transito (ricevuti da una coda da un consumatore, ma non ancora eliminati dalla coda). Se si raggiunge questa quota durante l'utilizzo del polling breve, Amazon SQS restituisce il messaggio di errore `OverLimit` . Se si utilizza un long polling, Amazon SQS non restituisce alcun messaggio di errore. Per evitare di raggiungere la quota, elimina i messaggi dalla coda dopo che sono stati elaborati. Puoi anche aumentare il numero di code utilizzate per elaborare i messaggi. Per richiedere un incremento della quota, invia una richiesta di supporto.
Nome coda	Un nome della coda può avere fino a 80 caratteri. I seguenti caratteri sono accettati: caratteri alfanumerici, trattini (`-`) e trattini bassi (`_`). **Note** I nomi coda fanno distinzione tra maiuscole e minuscole (ad esempio, `Test-queue` e `test-queue` sono code diverse).
Tag coda	Non è consigliabile aggiungere più di 50 tag a una coda. I tag supportano i caratteri Unicode in formato UTF-8.
	Il tag `Key` è obbligatorio, ma il tag `Value` è facoltativo.

Quota	Descrizione
	I tag `Key` e tag `Value` fanno distinzione tra maiuscole e minuscole.
	I tag `Key` e `Value` possono includere caratteri alfanumer ici Unicode in UTF-8 e spazi. I seguenti caratteri speciali sono consentiti: `_ . : / = + - @`
	Il tag `Key` o `Value` non deve includere il prefisso riservato `aws:` (non puoi eliminare chiavi o valori di tag con questo prefisso).
	La lunghezza massima del tag `Key` è 128 caratteri Unicode in UTF-8. Il tag `Key` non deve essere vuoto o nullo.
	La lunghezza massima del tag `Value` è 256 caratteri Unicode in UTF-8. Il tag `Value` può essere vuoto o nullo.
	Le azioni di tagging sono limitate a 30 TPS per Account AWS. Se la tua applicazione richiede una velocità di trasmissione effettiva più elevata, invia una richiesta.

Nozioni di base sulle code FIFO di Amazon SQS

Le code FIFO (First-In-First-Out) hanno tutte le capacità delle code standard, ma sono progettate per migliorare la messaggistica tra le applicazioni quando l'ordine delle operazioni e degli eventi è fondamentale, o laddove un duplicato non può essere tollerato.

Di seguito sono riportati alcuni esempi di situazioni in cui è possibile utilizzare le code FIFO:

- Sistema di gestione degli ordini di e-commerce in cui l'ordine è fondamentale
- Integrazione con sistemi di terze parti in cui gli eventi devono essere elaborati in ordine
- Elaborazione degli input inseriti dall'utente nell'ordine inserito
- Comunicazioni e reti: invio e ricezione di dati e informazioni nello stesso ordine
- Sistemi informatici: garanzia che i comandi immessi dall'utente vengano eseguiti nell'ordine corretto
- Istituti di istruzione: impedire a uno studente di iscriversi a un corso prima di registrare un account.
- Sistema di biglietteria online: dove i biglietti vengono distribuiti in base al principio “primo arrivato, primo servito"

> **Note**
>
> Le code FIFO forniscono inoltre l'elaborazione exactly-once, ma dispongono di un numero limitato di transazioni al secondo (TPS): Puoi utilizzare la modalità velocità di trasmissione effettiva elevata di Amazon SQS con la coda FIFO per aumentare il limite di transazioni. Per dettagli sull'utilizzo della modalità velocità di trasmissione effettiva elevata, consulta Velocità di trasmissione effettiva elevata per le code FIFO. Per ulteriori informazioni sulle quote di velocità di trasmissione effettiva, vedere the section called “Quote correlate ai messaggi”.

Le code FIFO di Amazon SQS sono disponibili in tutte le regioni in cui è disponibile Amazon SQS.

Per ulteriori informazioni sull'utilizzo delle code FIFO con ordinamenti complessi, consulta Soluzione delle sfide di ordinamento complesse con le code FIFO di Amazon SQS.

Per informazioni su come creare e configurare code utilizzando la console Amazon SQS, consulta Creazione di una coda (console). Per esempi in Java, consulta Esempi di SDK Java di Amazon SQS.

Per le best practice relative all'utilizzo di code FIFO, consulta Consigli aggiuntivi per le code FIFO di Amazon SQS e Consigli per le code FIFO e standard di Amazon SQS.

Logica di distribuzione FIFO

Le seguenti nozioni possono aiutarti a comprendere meglio l'invio e la ricezione di messaggi da FIFO.

Invio di messaggi

Se più messaggi vengono inviati in successione a una coda FIFO, ognuno con un ID di deduplicazione messaggio distinto, Amazon SQS archivia i messaggi e riconosce la trasmissione. Ogni messaggio può quindi essere ricevuto ed elaborato nell'ordine esatto in cui i messaggi sono stati trasmessi.

Nelle code FIFO, i messaggi vengono ordinati in base all'ID del gruppo messaggi. Se più host (o diversi thread sullo stesso host) inviano messaggi con lo stesso ID gruppo di messaggi a una coda FIFO, Amazon SQS memorizza i messaggi nell'ordine in cui arrivano per l'elaborazione. Per garantire che Amazon SQS conservi l'ordine in cui i messaggi vengono inviati e ricevuti, accertarsi che ogni produttore utilizzi un ID gruppo di messaggi univoco per inviare tutti i suoi messaggi.

La logica della coda FIFO si applica solo all'ID del gruppo di messaggi. Ogni ID del gruppo di messaggi rappresenta un gruppo di messaggi ordinati distinto all'interno di una coda Amazon SQS. Per ciascun ID gruppo messaggi, tutti i messaggi vengono inviati e ricevuti in un ordine esatto. Tuttavia, i messaggi con valori ID gruppo messaggi diversi possono essere inviati e ricevuti fuori ordine. Devi associare un ID gruppo di messaggi a un messaggio. Se non fornisci un ID gruppo di messaggi, l'operazione non riesce. Se hai bisogno di un singolo gruppo di messaggi ordinati, fornisci lo stesso ID gruppo messaggi per i messaggi inviati alla coda FIFO.

Ricezione di messaggi

Non puoi richiedere di ricevere messaggi con un ID gruppo messaggio specifico.

Quando ricevi messaggi da una coda FIFO con più ID gruppo di messaggi, Amazon SQS tenta prima di restituire quanti più messaggi con lo stesso ID gruppo di messaggi possibile. In questo modo altri consumatori possono elaborare messaggi con un altro ID gruppo messaggi. Quando ricevi un messaggio con un ID gruppo di messaggi, non vengono restituiti altri messaggi per lo stesso ID gruppo di messaggi a meno che non elimini il messaggio o non diventi visibile.

> **Note**
>
> È possibile ricevere fino a 10 messaggi in una singola chiamata utilizzando il parametro di richiesta `MaxNumberOfMessages` dell'operazione `ReceiveMessage`. Questi messaggi

mantengono il loro ordine FIFO e possono avere lo stesso ID messaggio di gruppo. Pertanto, se sono presenti meno di 10 messaggi disponibili con lo stesso ID messaggio di gruppo, potresti ricevere messaggi da un altro ID messaggio di gruppo, nello stesso batch di 10 messaggi, ma ancora in ordine FIFO.

Tentativi di riesecuzione multipli

Le code FIFO consentono al produttore o al consumatore di fare più tentativi:

- Se il produttore rileva un'azione `SendMessage` non riuscita, può ritentare l'invio tutte le volte che è necessario, utilizzando lo stesso ID di deduplicazione del messaggio. Supponendo che il produttore riceva almeno una conferma prima della scadenza dell'intervallo di deduplicazione, più tentativi non influiscono sull'ordine dei messaggi né introducono duplicati.
- Se il consumatore rileva un'azione `ReceiveMessage` non riuscita, può riprovare tutte le volte che è necessario, utilizzando lo stesso ID del tentativo di richiesta di ricezione. Supponendo che il consumatore riceva almeno una conferma prima della scadenza del timeout di visibilità, più tentativi non influiscono sull'ordine dei messaggi.
- Quando ricevi un messaggio con un ID gruppo di messaggi, non vengono restituiti altri messaggi per lo stesso ID gruppo di messaggi a meno che non elimini il messaggio o non diventi visibile.

Ordine dei messaggi

La coda FIFO migliora e integra la coda standard. Le caratteristiche più importanti di questo tipo di coda sono consegna FIFO (First-In-First-Out) ed elaborazione exactly-once:

- L'ordine in cui i messaggi vengono inviati e ricevuti viene rigorosamente mantenuto e un messaggio viene consegnato una sola volta e non è disponibile finché un consumatore non lo elabora ed elimina.
- I duplicati non vengono introdotti nella coda.

Inoltre, le code FIFO supportano gruppi di messaggi che consentono gruppi di messaggi ordinati multipli all'interno di una singola coda. Non è previsto alcun limite al numero di gruppi di messaggi all'interno di una coda FIFO.

Elaborazione "exactly-once"

A differenza delle code standard, le code FIFO non introducono messaggi duplicati. Le code FIFO consentono di evitare l'invio di un duplicato a una coda. Se ritenti l'operazione `SendMessage` entro l'intervallo di deduplicazione di 5 minuti, Amazon SQS non introduce duplicati nella coda.

Per configurare la deduplicazione, devi procedere in uno dei seguenti modi:

- Abilitare la deduplicazione basata sui contenuti. Questo istruisce Amazon SQS a utilizzare un hash SHA-256 per generare l'ID di deduplicazione messaggio utilizzando il corpo del messaggio, ma non gli attributi dello stesso. Per ulteriori informazioni, consulta la documentazione sulle operazioni `CreateQueue`, `GetQueueAttributes` e `SetQueueAttributes` nella Guida di riferimento delle API Amazon Simple Queue Service.
- Offrire in modo esplicito l'ID di deduplicazione messaggio (o visualizzare il numero di sequenza) per il messaggio. Per ulteriori informazioni, consulta la documentazione sulle operazioni `SendMessage`, `SendMessageBatch` e `ReceiveMessage` nella Guida di riferimento delle API Amazon Simple Queue Service.

Spostamento da una coda standard a una coda FIFO

Se disponi di un'applicazione esistente che utilizza code standard e desideri sfruttare delle caratteristiche di ordinamento o elaborazione "exactly-once" delle code FIFO, devi configurare la coda e l'applicazione correttamente.

> **Note**
>
> Non puoi convertire una coda standard esistente un una coda FIFO. Dovrai invece creare una nuova coda FIFO per la tua applicazione o eliminare la coda standard esistente e ricrearla come coda FIFO.

Utilizza la seguente checklist per assicurarti che l'applicazione funzioni correttamente con una coda FIFO.

- Utilizza la modalità a velocità di trasmissione effettiva elevata consigliata per FIFO per ottenere una maggiore velocità di trasmissione efefttiva. Per ulteriori informazioni sui limiti dei messaggi, vedere Quote correlate ai messaggi.

- Le code FIFO non supportano i ritardi per messaggio, ma solo i ritardi per coda. Se la tua applicazione imposta lo stesso valore del parametro `DelaySeconds` su ogni messaggio, devi modificare la tua applicazione per rimuovere il ritardo per messaggio e impostare `DelaySeconds` sull'intera coda.
- Il gruppo di messaggi è una funzionalità FIFO unica che consente ai clienti di elaborare i messaggi in parallelo mantenendo i rispettivi ordini. I clienti organizzano i messaggi in gruppi di messaggi specificando un ID del gruppo di messaggi. I gruppi di messaggi si basano spesso su una dimensione aziendale per un determinato carico di lavoro. Per una migliore scalabilità con le code FIFO, utilizza una dimensione aziendale più granulare per l'ID dei messaggi. Maggiore è il numero di ID dei gruppi di messaggi a cui distribuisci i messaggi, maggiore è il numero di messaggi che FIFO rende disponibili per l'utilizzo.
- Prima di inviare messaggi a una coda FIFO, conferma quanto segue:
  - Se la tua applicazione è in grado di inviare messaggi con lo stesso corpo, puoi modificare la tua applicazione per fornire un ID di deduplicazione dei messaggi univoco per ogni messaggio inviato.
  - Se la tua applicazione invia messaggi con corpi univoci, puoi abilitare la deduplicazione basata su contenuto.
- Non devi effettuare modifiche al codice per il tuo consumatore. Tuttavia, se occorre troppo tempo per elaborare i messaggi e il timeout visibilità è impostato su un valore elevato, puoi aggiungere un ID tentativo di richiesta di ricezione per ciascuna operazione `ReceiveMessage`. Ciò consente di riprovare tentativi di ricezione in caso di guasti di rete ed evita che le code vengano sospese a causa di tentativi di ricezione non riusciti.

Per ulteriori informazioni, consulta Guida di riferimento per l'API di Amazon Simple Storage Service.

Velocità di trasmissione effettiva elevata per le code FIFO

L'elevata velocità di trasmissione effettiva per le code FIFO supporta un numero maggiore di richieste per API, al secondo. Per aumentare il numero di richieste nella velocità di trasmissione effettiva elevata per le code FIFO, puoi aumentare il numero di gruppi di messaggi che utilizzi. Per ulteriori informazioni sulle quote di messaggi ad alta velocità di trasmissione effettiva, consulta Quote del servizio Amazon SQS nella Riferimenti generali di Amazon Web Services. Per informazioni sulle quote per coda con velocità di trasmissione effettiva elevata per le quote FIFO, vedere Quote correlate ai messaggi e Partizioni e distribuzione dei dati per una velocità di trasmissione effettiva elevata per le code FIFO SQS.

Argomenti

Partizioni e distribuzione dei dati per una velocità di trasmissione effettiva elevata per le code FIFO SQS

Amazon SQS memorizza i dati delle code FIFO nelle partizioni. Una partizione è un'allocazione di archiviazione per una coda replicata automaticamente su più zone di disponibilità all'interno di una regione AWS. Non si gestiscono le partizioni. Al contrario, Amazon SQS gestisce la gestione delle partizioni.

Per le code FIFO, Amazon SQS modifica il numero di partizioni in una coda nelle seguenti situazioni:

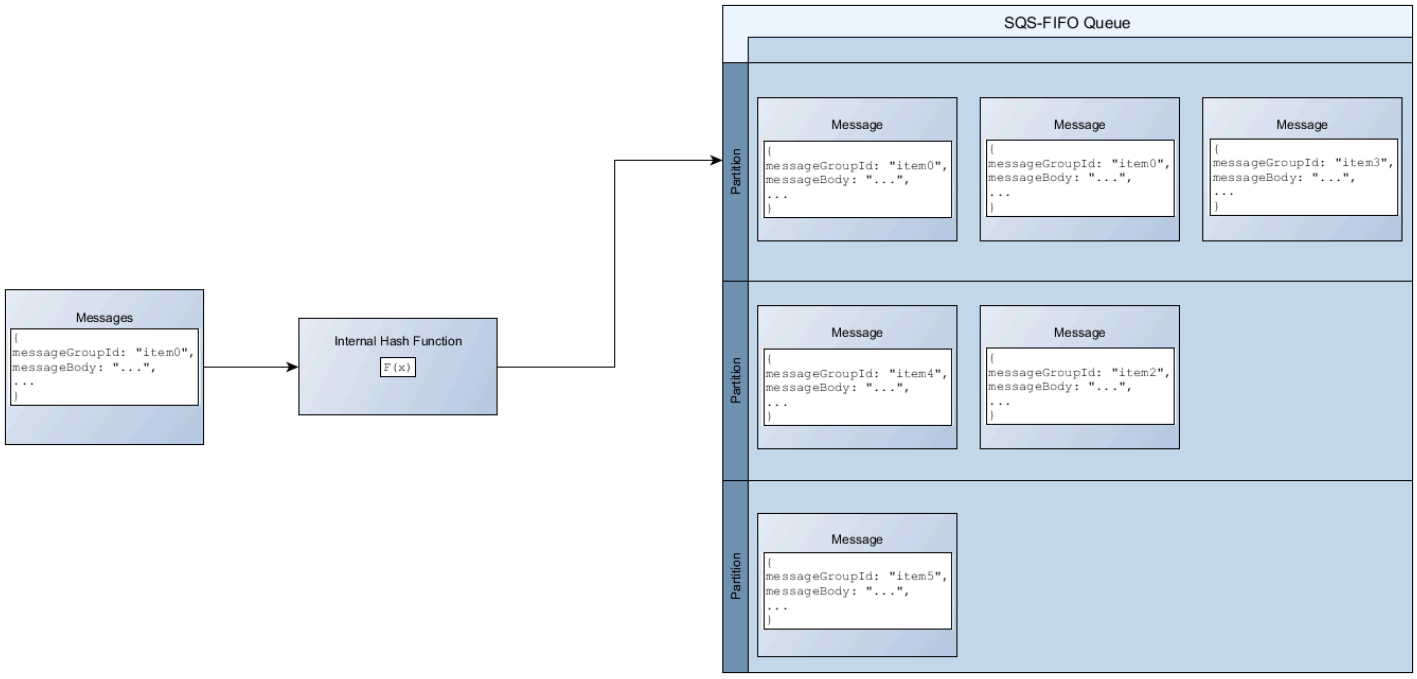
- Se la frequenza di richiesta corrente si avvicina o supera quella supportata dalle partizioni esistenti, vengono allocate partizioni aggiuntive finché la coda non raggiunge la quota regionale. Per informazioni sulle quote, consulta Quote correlate ai messaggi.
- Se le partizioni correnti sono poco utilizzate, il numero di partizioni può essere ridotto.

La gestione delle partizioni avviene automaticamente in background ed è trasparente per le tue applicazioni. La coda e i messaggi sono sempre disponibili.

Distribuzione dei dati per ID dei gruppi di messaggi

Per aggiungere un messaggio a una coda FIFO, Amazon SQS utilizza il valore dell'ID del gruppo di messaggi di ogni messaggio come input per una funzione hash interna. Il valore di output dalla funzione hash determina la partizione in cui verrà memorizzato il messaggio.

Il seguente diagramma mostra una coda che si estende su più partizioni. L'ID del gruppo di messaggi della coda si basa sul numero dell'elemento. Amazon SQS utilizza la funzione hash per determinare dove memorizzare un nuovo elemento, in questo caso in base al valore hash della stringa `item0`. Si noti che gli elementi vengono memorizzati nello stesso ordine in cui vengono aggiunti alla coda. La posizione di ciascun item è determinata dal valore hash dell'ID del gruppo di messaggi.



> **Note**
>
> Amazon SQS è ottimizzato per una distribuzione uniforme degli elementi sulle partizioni di una coda FIFO, indipendentemente dal numero di partizioni. AWS consiglia di utilizzare ID di gruppi di messaggi che possono avere un gran numero di valori distinti.

Ottimizzazione dell'utilizzo delle partizioni

Ogni partizione supporta fino a 3.000 messaggi al secondo con batch o fino a 300 messaggi al secondo per operazioni di invio, ricezione ed eliminazione nelle regioni supportate. Per ulteriori informazioni sulle quote di messaggi ad alta velocità di trasmissione effettiva, consulta Quote del servizio Amazon SQS nella Riferimenti generali di Amazon Web Services.

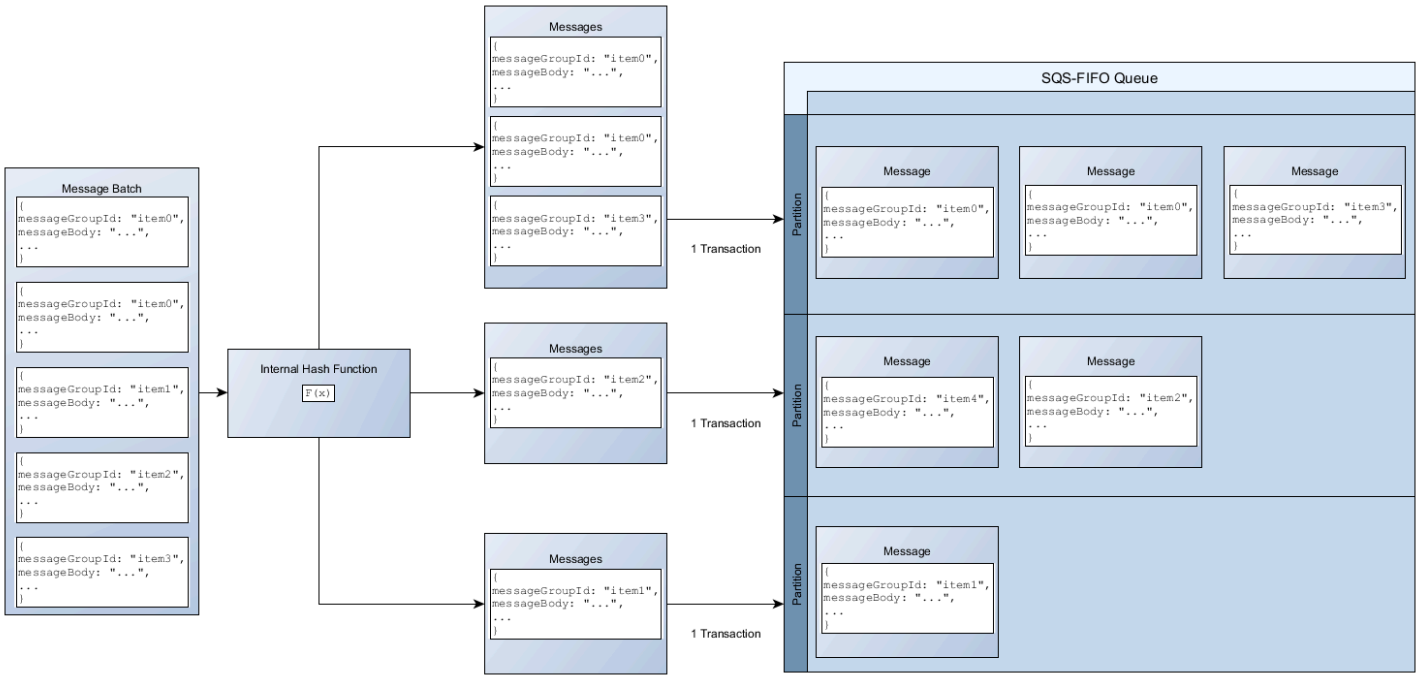
Quando si utilizzano API in batch, ogni messaggio viene instradato in base al processo descritto in Distribuzione dei dati per ID dei gruppi di messaggi. I messaggi indirizzati alla stessa partizione vengono raggruppati ed elaborati in un'unica transazione.

Per ottimizzare l'utilizzo delle partizioni per l'API `SendMessageBatch`, AWS consiglia di raggruppare i messaggi in batch con gli stessi ID di gruppo di messaggi, quando possibile.

Per ottimizzare l'utilizzo delle partizioni per le API `DeleteMessageBatch` e `ChangeMessageVisibilityBatch`, AWS consiglia di utilizzare le richieste `ReceiveMessage`

con il parametro `MaxNumberOfMessages` impostato su 10 e di raggruppare in batch gli handle di ricezione restituiti da una singola richiesta `ReceiveMessage`.

Nell'esempio seguente, viene inviato un batch di messaggi con diversi ID di gruppi di messaggi. Il batch viene suddiviso in tre gruppi, ognuno dei quali viene conteggiato nella quota della partizione.



> **Note**
>
> Amazon SQS garantisce solo che i messaggi con la funzione hash interna dello stesso ID del gruppo di messaggi siano raggruppati all'interno di una richiesta batch. A seconda dell'output della funzione hash interna e del numero di partizioni, è possibile raggruppare messaggi con ID di gruppi di messaggi diversi. Poiché la funzione hash o il numero di partizioni possono cambiare in qualsiasi momento, i messaggi raggruppati in un punto non possono essere raggruppati in un secondo momento.

Abilita la velocità di trasmissione effettiva elevata per le code FIFO

Puoi abilitare una velocità di trasmissione effettiva elevata per qualsiasi coda FIFO nuova o esistente. La funzionalità include tre nuove opzioni per la creazione e la modifica delle code FIFO:

- Abilita FIFO a velocità di trasmissione effettiva elevata: da utilizzare per abilitare una velocità di trasmissione effettiva elevata per i messaggi nella coda FIFO attuale.

- Ambito di deduplicazione: specifica se la deduplicazione avviene a livello di coda o di gruppo di messaggi.
- Limite di velocità effettiva FIFO: specifica se la quota di velocità effettiva per i messaggi nella coda FIFO è impostata a livello di coda o di gruppo di messaggi.

Per abilitare una velocità di trasmissione effettiva elevata per una coda FIFO (console)

1. Inizia a creare o modificare una coda FIFO.
2. Quando si specificano le opzioni per la coda, scegliere Abilita FIFO ad alta velocità di trasmissione effettiva.

   L'abilitazione di una velocità di trasmissione effettiva elevata per le code FIFO imposta le opzioni correlate come segue:

   - Ambito di deduplicazione è impostato su Gruppo di messaggi, l'impostazione richiesta per l'utilizzo di una velocità di trasmissione effettiva elevata per le code FIFO.
   - Limite FIFO di velocità di trasmissione effettiva è impostato su ID gruppo di messaggi, l'impostazione richiesta per l'utilizzo di una velocità di trasmissione effettiva elevata per le code FIFO.

   Se si modifica una delle impostazioni necessarie per utilizzare le code FIFO ad alta velocità di trasmissione effettiva, si applica la velocità di trasmissione effettiva normale per la coda e la deduplicazione si verifica come specificato.
3. Continuare a specificare tutte le opzioni per la coda. Al termine, scegliere Crea coda o Salva.

Dopo aver creato o modificato la coda FIFO, puoi inviarle messaggi e riceverli ed eliminarli, il tutto con un TPS più elevato. Per quote di velocità di trasmissione effettiva elevata, consulta Velocità di trasmissione effettiva dei messaggi in Quote correlate ai messaggi.

Termini chiave

I seguenti termini chiave possono aiutarti a comprendere meglio le funzionalità delle code FIFO. Per ulteriori informazioni, consulta la Guida di riferimento per l'API di Amazon Simple Storage Service.

ID deduplicazione messaggi

Il token utilizzato per la deduplicazione dei messaggi inviati. Se un messaggio con un particolare ID di deduplicazione del messaggio viene inviato con successo, tutti i messaggi inviati con lo stesso ID di deduplicazione del messaggio vengono accettati correttamente, ma non vengono recapitati entro l'intervallo di deduplicazione di 5 minuti.

> **Note**
>
> Amazon SQS continua a tenere traccia dell'ID di deduplicazione dei messaggi anche dopo che il messaggio viene ricevuto ed eliminato.

ID gruppo di messaggi

Tag che specifica che un messaggio appartiene a un gruppo di messaggi specifico. I messaggi che appartengono allo stesso gruppo di messaggi vengono sempre elaborati uno per uno, in un ordine rigoroso rispetto al gruppo di messaggi (tuttavia, i messaggi che appartengono a gruppi di messaggi diversi potrebbero essere elaborati in modo errato).

ID tentativo richiesta di ricezione

Il token utilizzato per la deduplicazione delle chiamate `ReceiveMessage`.

Numero sequenza

Il numero grande e non consecutivo che Amazon SQS assegna a ciascun messaggio.

Compatibilità

Client

L'Amazon SQS Buffered Asynchronous Client attualmente non supporta le code FIFO.

Servizi

Se l'applicazione utilizza più AWS servizi o una combinazione di AWS servizi esterni, è importante capire quali funzionalità del servizio non supportano le code FIFO.

Alcuni servizi AWS o servizi esterni che inviano notifiche ad Amazon SQS potrebbero non essere compatibili con le code FIFO, nonostante consentano di impostare una coda FIFO come destinazione.

Le seguenti funzionalità dei AWS servizi non sono attualmente compatibili con le code FIFO:

- Notifiche di eventi Amazon S3
- Hook del ciclo di vita del dimensionamento automatico
- AWS IoT Azioni relative alle regole
- Code DLQ AWS Lambda

Per informazioni sulla compatibilità di altri servizi con le code FIFO, consulta la documentazione del servizio.

Identificatori di code e messaggi Amazon SQS

Questa sezione descrive gli identificatori delle code FIFO. Questi identificatori possono aiutarti a trovare e modificare code e messaggi specifici.

Argomenti

Identificatori per le code FIFO di Amazon SQS

Per ulteriori informazioni, consulta i seguenti argomenti nella Documentazione di riferimento delle API di Amazon Simple Queue Service.

Nome e URL della coda

Quando crei una nuova coda, è necessario specificare un nome coda univoco per il tuo account e la tua regione AWS . Amazon SQS assegna a ogni coda che crei un identificatore chiamato URL coda che include il nome della coda e altri componenti Amazon SQS. Se desideri eseguire un'operazione su una coda, devi fornire il relativo URL coda.

Il nome di una coda FIFO deve terminare con il suffisso `.fifo`. Il suffisso viene conteggiato ai fini della quota di 80 caratteri dei nomi della coda. Per determinare se una coda è FIFO, puoi verificare se il nome della coda termina con il suffisso.

Di seguito è riportato l'URL della coda per una coda FIFO denominata di `MyQueue` proprietà di un utente con il numero di account AWS. `123456789012`

```
https://sqs.us-east-2.amazonaws.com/123456789012/MyQueue.fifo
```

È possibile recuperare l'URL di una coda a livello di codice elencando le code e analizzando la stringa che segue il numero di account. Per ulteriori informazioni, consulta `ListQueues`.

ID messaggio

Ogni messaggio riceve un ID messaggio assegnato dal sistema che Amazon SQS restituisce nella risposta `SendMessage`. Questo identificatore è utile per l'identificazione di messaggi. La durata massima di un ID messaggio è di 100 caratteri.

Handle di ricezione

Ogni volta che ricevi un messaggio da una coda, ricevi un handle di ricezione per tale messaggio. Questo handle è associato all'operazione di ricezione del messaggio, non al messaggio stesso. Per eliminare il messaggio o per modificarne la visibilità, devi fornire l'handle di ricezione (non l'ID messaggio). Pertanto, devi sempre ricevere un messaggio prima di poterlo eliminare (non puoi inserire un messaggio nella coda e poi richiamarlo). La durata massima di un handle di ricezione è di 1024 caratteri.

> **Important**
>
> Se ricevi un messaggio più di una volta, ogni volta che lo ricevi ottieni un diverso handle di ricezione. Devi fornire l'handle di ricezione ricevuto più di recente quando richiedi di eliminare il messaggio (in caso contrario, il messaggio potrebbe non essere eliminato).

Di seguito è riportato un esempio di handle di ricezione (suddiviso su tre linee).

```
MbZj6wDWli+JvwwJaBV+3dcjk2YW2vA3+STFFljTM8tJJg6HRG6PYSasuWXPJB+Cw
Lj1FjgXUv1uSj1gUPAWV66FU/WeR4mq2OKpEGYWbnLmpRCJVAyeMjeU5ZBdtcQ+QE
auMZc8ZRv37sIW2iJKq3M9MFx1YvV11A2x/KSbkJ0=
```

Identificatori aggiuntivi per le code FIFO di Amazon SQS

Per ulteriori informazioni sui seguenti identificatori, consulta gli argomenti Elaborazione "exactly-once" e la Documentazione di riferimento delle API di Amazon Simple Queue Service.

ID deduplicazione messaggi

Il token utilizzato per la deduplicazione dei messaggi inviati. Se un messaggio con un particolare ID di deduplicazione del messaggio viene inviato con successo, tutti i messaggi inviati con lo stesso ID di deduplicazione del messaggio vengono accettati correttamente, ma non vengono recapitati entro l'intervallo di deduplicazione di 5 minuti.

ID gruppo di messaggi

Tag che specifica che un messaggio appartiene a un gruppo di messaggi specifico. I messaggi che appartengono allo stesso gruppo di messaggi vengono sempre elaborati uno per uno, in un ordine rigoroso rispetto al gruppo di messaggi (tuttavia, i messaggi che appartengono a gruppi di messaggi diversi potrebbero essere elaborati in modo errato).

Numero sequenza

Il numero grande e non consecutivo che Amazon SQS assegna a ciascun messaggio.

Quote

La tabella seguente elenca le quote correlate alle code FIFO.

Quota	Descrizione
Coda di ritardo	Il ritardo predefinito (minimo) per una coda è 0 secondi. Il valore massimo è 15 minuti.
Code elencate	1.000 code per ogni richiesta `ListQueues` .
Tempo di attesa del long polling	Il tempo massimo di attesa per il long polling è di 20 secondi.
Gruppi di messaggi	Non è previsto alcun limite al numero di gruppi di messaggi all'interno di una coda FIFO.
Messaggi per coda (backlog)	Il numero di messaggi che una coda Amazon SQS può archiviare è illimitato.
Messaggi per coda (in transito)	Per le code FIFO, possono esserci un massimo di 20.000 messaggi in corso (ricevuti da una coda da un

Quota	Descrizione
	consumatore, ma non ancora eliminati dalla coda). Se raggiungi questa quota, Amazon SQS non restituisce alcun messaggio di errore.
Nome coda	Il nome di una coda FIFO deve terminare con il suffisso `.fifo`. Il suffisso viene conteggiato ai fini della quota di 80 caratteri dei nomi della coda. Per determinare se una coda è FIFO, puoi verificare se il nome della coda termina con il suffisso.
Tag coda	Non è consigliabile aggiungere più di 50 tag a una coda. I tag supportano i caratteri Unicode in formato UTF-8.
	Il tag `Key` è obbligatorio, ma il tag `Value` è facoltativo.
	I tag `Key` e tag `Value` fanno distinzione tra maiuscole e minuscole.
	I tag `Key` e `Value` possono includere caratteri alfanumer ici Unicode in UTF-8 e spazi. I seguenti caratteri speciali sono consentiti: `_ . : / = + - @`
	Il tag `Key` o `Value` non deve includere il prefisso riservato `aws:` (non puoi eliminare chiavi o valori di tag con questo prefisso).
	La lunghezza massima del tag `Key` è 128 caratteri Unicode in UTF-8. Il tag `Key` non deve essere vuoto o nullo.
	La lunghezza massima del tag `Value` è 256 caratteri Unicode in UTF-8. Il tag `Value` può essere vuoto o nullo.
	Le azioni di tagging sono limitate a 30 TPS ciascuna. Account AWS Se la tua applicazione richiede una velocità di trasmissione effettiva più elevata, invia una richiesta.

Quote di Amazon SQS

Questo argomento elenca le quote all'interno di Amazon Simple Queue Service (Amazon SQS).

Argomenti

Quote correlate ai messaggi

La tabella seguente elenca le quote relative ai messaggi.

Quota	Descrizione
ID messaggio con batch	Un ID messaggio con batch può contenere fino a 80 caratteri. I seguenti caratteri sono accettati: caratteri alfanumerici, trattini (-) e trattini bassi (_).
Attributi di messaggio	Un messaggio può contenere fino a 10 attributi di metadati.
Batch di messaggi	Una singola richiesta di batch di messaggi può includere un massimo di 10 messaggi. Per ulteriori informazi oni, consulta Configurazione di AmazonSQSBufferedA syncClient nella sezione Operazioni in batch per Amazon SQS.
Contenuto del messaggio	Un messaggio può includere solo in formato JSON, XML e testo non formattato. I seguenti caratteri Unicode sono consentiti: `#x9` | `#xA` | `#xD` | da `#x20` a `#xD7FF` | da `#xE000` a `#xFFFD` | da `#x10000` a `#x10FFFF` I caratteri non inclusi in questo elenco vengono rifiutati. Per maggiori informazioni consulta le specifiche W3C per i caratteri.

Quota	Descrizione
ID gruppo di messaggi	Utilizzare i messaggi dal backlog per evitare di creare un backlog di messaggi di grandi dimensioni con lo stesso ID gruppo di messaggi. `MessageGroupId` è obbligatorio per le code FIFO. Non puoi utilizzarlo per code standard. È necessario associare un `MessageGroupId` non vuoto a un messaggio. Se non fornisci un `MessageGroupId`, l'azione non può essere completata. La lunghezza massima di `MessageGroupId` è 128 caratteri. Valori validi: caratteri alfanumerici e punteggiatura `(!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~)`.
Conservazione dei messaggi	Per impostazione predefinita, un messaggio viene conservato per 4 giorni. Il minimo è di 60 secondi (1 minuto). Il massimo è 1.209.600 secondi (14 giorni).
Throughput dei messaggi	Le code standard supportano un numero quasi illimitato di chiamate API al secondo, per azione API (`SendMessage`, `ReceiveMessage` o `DeleteMessage`).
	Code FIFO • Le code FIFO supportano una quota di 300 transazioni al secondo per azione API (`SendMessage`, `ReceiveMessage` e `DeleteMessage`). • Se si utilizza il batch, le code FIFO supportano fino a 3.000 messaggi al secondo per azione API (`SendMessage`, `ReceiveMessage` o `DeleteMessage`). I 3000 messaggi rappresentano 300 chiamate API, ognuna con un batch di 10 messaggi. Per richiedere un incremento della quota, invia una richiesta di supporto.

Quota	Descrizione
	Velocità di trasmissione effettiva elevata per le code FIFO • Senza raggruppamenti in batch (`SendMessage` , `ReceiveMessage` e `DeleteMessage`), la velocità di trasmissione effettiva elevata per le code FIFO elabora fino a 70.000 transazioni al secondo per azione API nelle regioni Stati Uniti orientali (Virginia settentri onale), Stati Uniti occidentali (Oregon) ed Europa (Irlanda). • Per le regioni Stati Uniti orientali (Ohio) ed Europa (Francoforte), la velocità di trasmissione effettiva predefinita è di 18.000 transazioni al secondo per azione API. • Per le regioni Asia Pacifico (Mumbai), Asia Pacifico (Singapore), Asia Pacifico (Sydney) e Asia Pacifico (Tokyo), la velocità effettiva predefinita è di 9.000 transazioni al secondo per operazione API. • Per Europa (Londra) e Sud America (San Paolo), la velocità effettiva predefinita è di 4.500 transazioni al secondo per azione API. • Per massimizzare la velocità di trasmissione effettiva, aumenta il numero di ID dei gruppi di messaggi utilizzat i per i messaggi inviati senza batch. • È possibile aumentare la velocità di trasmissione effettiva fino a 700.000 messaggi al secondo utilizzan do API in batch (`SendMessageBatch` e `DeleteMes sageBatch`) nelle regioni Stati Uniti orientali (Virginia settentrionale), Stati Uniti occidentali (Oregon) ed Europa (Irlanda). I 700.000 messaggi al secondo rappresentano 70.000 transazioni al secondo, ognuna con un batch di 10 messaggi. Per le regioni Europa (Francoforte) e Stati Uniti orientali (Ohio), è possibile ricevere fino a 180.000 messaggi

Quota	Descrizione
	al secondo utilizzando le API di batch. I 180.000 messaggi al secondo rappresentano 18.000 transazioni al secondo, ognuna con un batch di 10 messaggi. Per le regioni Asia Pacifico (Mumbai), Asia Pacifico (Singapore), Asia Pacifico (Sydney) e Asia Pacifico (Tokyo), è possibile inviare fino a 90.000 messaggi al secondo senza batch. Per ottenere la velocità di trasmissione effettiva massima quando si utilizza `SendMessageBatch` e `DeleteMessageBatch`, tutti i messaggi in una richiesta batch devono utilizzare lo stesso ID del gruppo di messaggi. • Per le regioni Europa (Londra) e Sud America (San Paolo), è possibile ricevere fino a 45.000 messaggi al secondo con il batching. Per ottenere la velocità di trasmissione effettiva massima quando si utilizza `SendMessageBatch` e `DeleteMessageBatch`, tutti i messaggi in una richiesta batch devono utilizzare lo stesso ID del gruppo di messaggi. • In tutte le altre AWS regioni, la velocità massima è di 2.400 (senza batch) o 24.000 (utilizzando il batch) messaggi al secondo, per azione API. • Per ulteriori informazioni, consulta Partizioni e distribuzione dei dati per una velocità di trasmissione effettiva elevata per le code FIFO SQS.
Timer messaggio	Il ritardo predefinito (minimo) per un messaggio è 0 secondi. Il valore massimo è 15 minuti.

Quota	Descrizione
Dimensione dei messaggi	La dimensione minima del messaggio è pari a 1 byte (1 carattere). La dimensione massima è 262.144 byte (256 KiB). Per inviare messaggi di dimensioni superiori a 256 KiB, puoi utilizzare Amazon SQS Extended Client Library per Java e Amazon SQS Extended Client Library for Python. Questa libreria consente di inviare un messaggio Amazon SQS che contiene un riferimento a un payload del messaggio in Amazon S3. La dimensione massima di payload è pari a 2 GB. **Note** Questa libreria estesa funziona solo per client sincroni.
Timeout visibilità del messaggio	Il timeout visibilità predefinito per una coda è di 30 secondi. Il valore minimo è 0 secondi. La durata massima è 12 ore.
Informazioni sulle policy	La quota massima è di 8.192 byte, 20 istruzioni, 50 principal o 10 condizioni. Per ulteriori informazioni, consulta Quote correlate alle policy.

Quote correlate alle policy

La tabella seguente elenca le quote relative alle policy.

Nome	Massimo
Byte	8,192
Condizioni	10

Nome	Massimo
Principali	50
Dichiarazioni	20
Operazioni per dichiarazione	7

Caratteristiche e funzionalità di Amazon SQS

Amazon SQS offre le seguenti caratteristiche e funzionalità.

Argomenti

Metadati dei messaggi

Puoi utilizzare gli attributi di messaggio per collegare metadati personalizzati ai messaggi Amazon SQS per le applicazioni. Puoi utilizzare gli attributi del sistema di messaggi per archiviare i metadati per altri servizi AWS, ad esempio AWS X-Ray.

Argomenti

Attributi messaggio di Amazon SQS

Amazon SQS può includere metadati strutturati (come time stamp, dati geospaziali, firme e identificatori) con i messaggi tramite gli attributi dei messaggi. Ogni messaggio può avere fino a

10 attributi. Gli attributi di messaggio sono facoltativi e separati dal corpo del messaggio (sebbene vengano inviati insieme a esso). Il tuo consumatore può utilizzare gli attributi di messaggio per gestire un messaggio in un determinato modo senza la necessità di elaborare prima il corpo del messaggio. Per ulteriori informazioni sull'invio di messaggi con attributi tramite la console Amazon SQS, consulta Invio di un messaggio con attributi (console).

> **Note**
>
> Non confondere gli attributi di messaggio con gli attributi del sistema di messaggi: mentre puoi utilizzare gli attributi di messaggio per collegare metadati personalizzati ai messaggi Amazon SQS per le applicazioni, puoi utilizzare gli attributi del sistema di messaggi per archiviare i metadati per altri servizi AWS, ad esempio AWS X-Ray.

Argomenti

Componenti attributo del messaggio

> **Important**
>
> Tutti i componenti di un attributo del messaggio sono inclusi nella limitazione delle dimensioni del messaggio di 256 KB.
> `Name`, `Type`, `Value` e il corpo del messaggio non devono essere vuoti o nulli.

Ogni attributo di messaggio è costituito dai seguenti elementi:

- Nome: il nome dell'attributo del messaggio può contenere i seguenti caratteri: `A-Z`, `a-z`, `0-9`, sottolineatura (_), trattino (`-`) e punto (`.`). Le restrizioni si applicano come segue:
  - Può contenere fino a 256 caratteri
  - Non può iniziare con `AWS.` o `Amazon.` (o qualsiasi variazione in maiuscole/minuscole)
  - Fa distinzione tra lettere maiuscole e minuscole
  - Deve essere univoco tra tutti i nomi di attributo per il messaggio

  - Non deve iniziare o finire con un punto
  - Non deve avere punti in una sequenza
- Tipo: il tipo di dati dell'attributo del messaggio. I tipi supportati includono `String`, `Number` e `Binary`. È anche possibile aggiungere informazioni personalizzate per qualsiasi tipo di dati. Il tipo di dati ha le stesse limitazioni del corpo del messaggio (per ulteriori informazioni, consulta `SendMessage` nella Guida di riferimento di Amazon Simple Queue Service API). Inoltre, si applicano le limitazioni seguenti:
  - Può contenere fino a 256 caratteri
  - Fa distinzione tra lettere maiuscole e minuscole
- Valore: il valore dell'attributo di messaggio. Per i tipi di dati `String`, i valori dell'attributo hanno le stesse restrizioni del corpo del messaggio.

Tipi di dati degli attributi di messaggio

I tipi di dati degli attributi di messaggio indicano a Amazon SQS il modo in cui gestire i valori degli attributi dei messaggi corrispondenti. Ad esempio, se il tipo è `Number`, Amazon SQS convalida i valori numerici.

Amazon SQS supporta i tipi di dati logici `String`, `Number` e `Binary` con etichette opzionali di tipi di dati personalizzati nel formato *`.custom-data-type`*

- Stringa: gli attributi `String` possono memorizzare testo Unicode utilizzando qualsiasi carattere XML valido.
- Numero: gli attributi di `Number` possono memorizzare valori numerici positivi o negativi. Un numero può avere fino a 38 cifre di precisione e può essere compreso tra 10^-128 e 10^+126.

  > **Note**
  >
  > Amazon SQS rimuove zero iniziali e finali.

- Binary: gli attributi di tipo binario possono archiviare qualsiasi tipo di dati binari, ad esempio dati compressi, dati crittografati o immagini.
- Personalizzato: per creare un tipo di dato personalizzato, aggiungi un'etichetta di tipo personalizzato a qualsiasi tipo di dato. Ad esempio:
  - `Number.byte`, `Number.short`, `Number.inte` `Number.float` possono aiutare a distinguere tra i tipi di numero.

- `Binary.gif` e `Binary.png` possono aiutare a distinguere tra tipi di file.

> **Note**
>
> Amazon SQS non interpreta, convalida oppure utilizza i dati aggiunti.
> L'etichetta di tipo personalizzato ha le stesse restrizioni del corpo del messaggio.

Calcolo del digest dei messaggi MD5 per gli attributi di messaggi

Se utilizzi AWS SDK for Java, puoi ignorare questa sezione. La classe `MessageMD5ChecksumHandler` di SDK per Java supporta i digest di messaggi MD5 per gli attributi dei messaggi Amazon SQS.

Se utilizzi l'API della query o uno degli SDK AWS che non supportano i digest di messaggi MD5 per gli attributi dei messaggi Amazon SQS, occorre utilizzare le seguenti linee guida per eseguire il calcolo del digest del messaggio MD5.

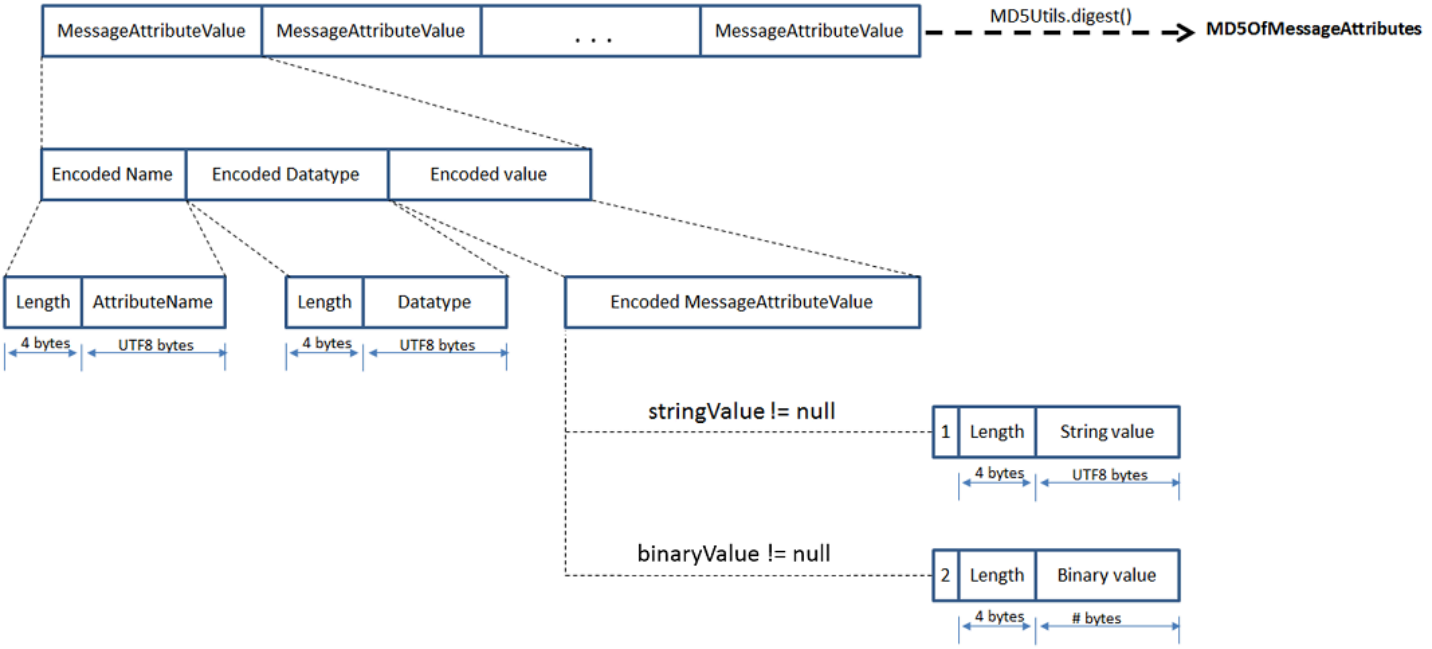
> **Note**
>
> Includi sempre suffissi di tipo di dati personalizzati nel calcolo del message-digest MD5.

Panoramica

La seguente è una panoramica dell'algoritmo di calcolo del digest dei messaggi MD5:

1. Ordinare tutti gli attributi dei messaggi in base al nome in ordine crescente.
2. Codificare le singole parti di ogni attributo (`Name`, `Type` e `Value`) in un buffer.
3. Calcolare il digest del messaggio dell'intero buffer.

Il seguente diagramma mostra la codifica del digest del messaggio MD5 per un singolo attributo del messaggio:



Per codificare un singolo attributo del messaggio Amazon SQS

1. Codificare il nome: lunghezza del nome (4 byte) e byte UTF-8 del nome.
2. Codificare il tipo di dati: lunghezza del nome (4 byte) e byte UTF-8 del tipo di dati.
3. Codificare il tipo di trasporto (`String` o `Binary`) del valore [1 byte].

   > **Note**
   >
   > I tipi di dati logici `String` e `Number` usano il tipo di trasporto `String`.
   > Il tipo di dati logici `Binary` utilizza il tipo di trasporto `Binary`.

   a. Per il tipo di trasporto `String`, codificare 1.

   b. Per il tipo di trasporto `Binary`, codificare 2.
4. Codificare il valore attributo.

   a. Per il tipo di trasporto `String`, codifica il valore di attributo: la lunghezza (4 byte) e i byte UTF-8 del valore.

   b. Per il tipo di trasporto `Binary`, codifica il valore di attributo: la lunghezza (4 byte) e i byte non elaborati del valore.

Attributi del sistema di messaggistica Amazon SQS

Mentre puoi utilizzare gli attributi di messaggio per collegare metadati personalizzati ai messaggi Amazon SQS per le applicazioni, puoi utilizzare gliattributi del sistema di messaggi per archiviare i metadati per altri servizi AWS, ad esempio AWS X-Ray. Per ulteriori informazioni, consulta il parametro di richiesta `MessageSystemAttribute` delle operazioni API `SendMessage` e `SendMessageBatch`, l'attributo `AWSTraceHeader` dell'operazione API `ReceiveMessage` e il tipo di dati `MessageSystemAttributeValue` nella Guida di riferimento di Amazon Simple Queue Service API.

Gli attributi del sistema di messaggi sono strutturati esattamente come gli attributi di messaggio, con le seguenti eccezioni:

- Al momento, l'unico attributo del sistema di messaggi supportato è `AWSTraceHeader`. Il tipo deve essere `String` e il valore deve essere una stringa di intestazione di traccia AWS X-Ray formattata correttamente.
- Le dimensioni di un attributo di sistema di messaggi non vengono conteggiate ai fini della dimensione totale di un messaggio.

Risorse necessarie per l'elaborazione di messaggi Amazon SQS

Per aiutarti a stimare le risorse di cui hai bisogno per elaborare messaggi in coda, Amazon SQS può determinare il numero approssimativo di messaggi differiti, visibili e non visibili in una coda. Per ulteriori informazioni sulla visibilità, consulta Timeout visibilità Amazon SQS.

> **Note**
>
> Per le code standard, il risultato è approssimativo a causa dell'architettura distribuita di Amazon SQS. Nella maggior parte dei casi, il conteggio deve essere vicino al numero effettivo di messaggi in coda.
> Per le code FIFO il risultato è esatto.

La tabella seguente elenca il nome attributo da usare con l'operazione `GetQueueAttributes`:

Attività	Nome attributo
Ottenere il numero approssimativo di messaggi disponibili per il recupero dalla coda.	`ApproximateNumberOfMessages Visible`
Ottenere il numero approssimativo di messaggi nella coda che vengono differiti e non sono disponibili per la lettura immediata. Ciò può accadere quando la coda è configurata come coda di ritardo o quando un messaggio è stato inviato con un parametro di ritardo.	`ApproximateNumberOfMessages Delayed`
Ottenere il numero approssimativo di messaggi che sono in transito. I messaggi sono considerati in transito se sono stati inviati a un client ma non sono ancora stati eliminati o non hanno ancora raggiunto il termine della loro finestra di visibilità.	`ApproximateNumberOfMessages NotVisible`

Elenca l'impaginazione delle code

I metodi `listQueues` e `listDeadLetterQueues` API supportano controlli di impaginazione opzionali. Per impostazione predefinita, questi metodi API restituiscono fino a 1000 code nel messaggio di risposta. È possibile impostare il parametro `MaxResults` in modo che restituisca un numero inferiore di risultati in ogni risposta.

Imposta il parametro `MaxResults` nella richiesta `listQueues` o `listDeadLetterQueues` per specificare il numero massimo di risultati da restituire nella risposta. Se non imposti `MaxResults`, la risposta include un massimo di 1.000 risultati e il valore `NextToken` nella risposta è nullo.

Se sono disponibili risultati aggiuntivi da visualizzare e imposti `MaxResults`, la risposta include un valore per `NextToken`. Utilizza `NextToken` come parametro nella richiesta successiva a `listQueues` per ricevere la pagina successiva dei risultati. Se il valore di `NextToken` nella risposta è nullo, non ci sono ulteriori risultati da visualizzare.

Tag per l'allocazione dei costi Amazon SQS

Per organizzare e identificare le code Amazon SQS per l'allocazione dei costi, puoi aggiungere tag metadati che identificano lo scopo, il proprietario o l'ambiente di una coda. Questo è particolarmente utile quando si dispone di numerose code. Per configurare i tag utilizzando la console di Amazon SQS, consulta the section called “Configurazione di tag per una coda”

Per organizzare le fatture AWS al fine di riflettere la struttura dei costi, puoi utilizzare i tag di allocazione dei costi. Per eseguire questa operazione, registrati per far sì che la fattura Account AWS del tuo account includa chiavi e valori di tag. Per ulteriori informazioni, consulta Impostazione di un report di allocazione dei costi mensili nella Guida per l'utente di AWS Billing.

Ogni tag è composto da una coppia chiave-valore definita dall'utente. Ad esempio, è possibile identificare facilmente le code di produzione e testing se tagghi le tue code nel modo seguente:

Queue	Chiave	Valore
`MyQueueA`	`QueueType`	`Production`
`MyQueueB`	`QueueType`	`Testing`

Note

Quando usi i tag di coda, tieni a mente le seguenti linee guida:

- Non è consigliabile aggiungere più di 50 tag a una coda. I tag supportano i caratteri Unicode in formato UTF-8.
- I tag non hanno alcun significato semantico. Amazon SQS interpreta i tag come stringhe di caratteri.
- I tag rispettano la distinzione tra maiuscole e minuscole.
- Un nuovo tag con una chiave identica a quella di un tag esistente sovrascrive il tag esistente.
- Le azioni di tagging sono limitate a 30 TPS per Account AWS. Se la tua applicazione richiede una velocità di trasmissione effettiva più elevata, invia una richiesta.

Per l'elenco completo dei tipi di risorse, consulta Quote.

Polling brevi e lunghi di Amazon SQS

Amazon SQS fornisce polling brevi e lunghi per ricevere messaggi da una coda. Per impostazione predefinita, le code utilizzano polling brevi.

Con polling breve, la richiesta interroga solo un sottoinsieme dei server (basato su una distribuzione casuale ponderata) per trovare i messaggi disponibili da includere nella risposta. Amazon SQS invia la risposta immediatamente, anche se la query non ha trovato messaggi.

In caso di polling prolungato, la richiesta `ReceiveMessage` richiede la ricerca di messaggi su tutti i server. Amazon SQS invia una risposta dopo aver raccolto almeno un messaggio disponibile, fino al numero massimo di messaggi specificato nella richiesta. Amazon SQS invia una risposta vuota solo se scade il tempo di attesa per il polling.

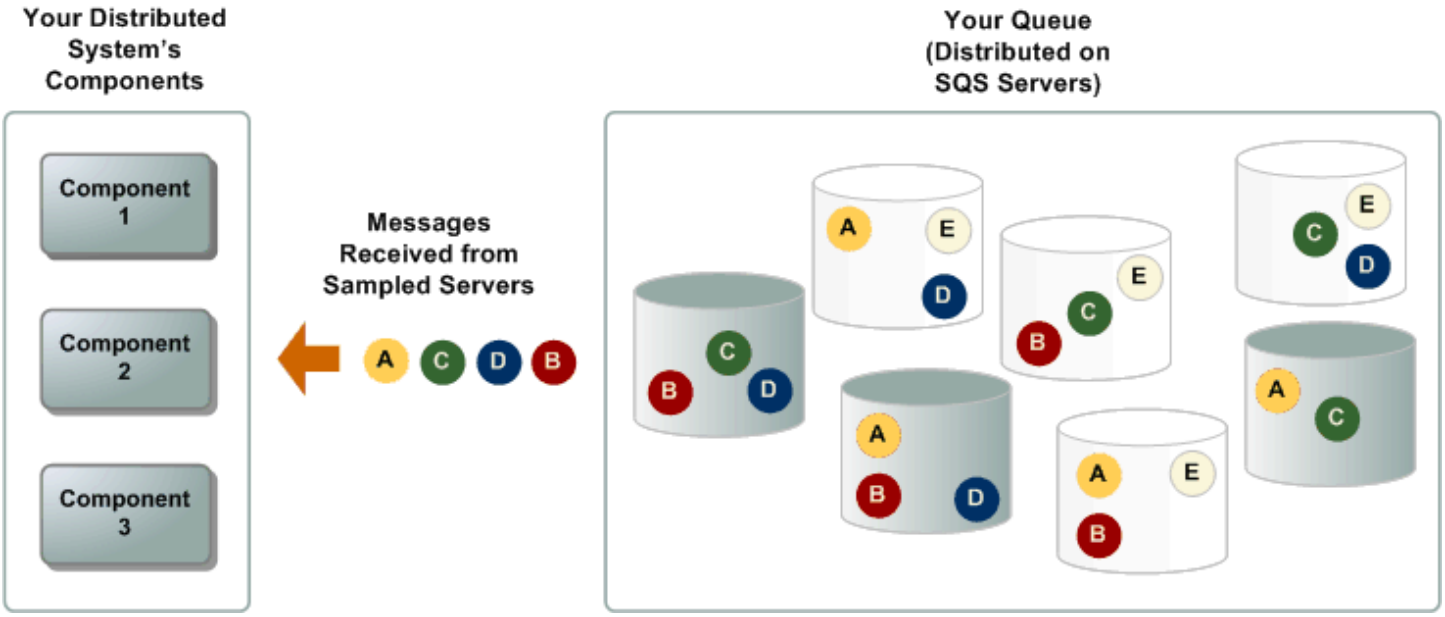
Nelle sezioni seguenti vengono illustrati i dettagli del polling breve e del polling lungo.

Argomenti

Utilizzo dei messaggi con lo short polling

Quando utilizzi messaggi da una coda con lo short breve, Amazon SQS campiona un sottoinsieme dei propri server (in base a una distribuzione random ponderata) e restituisce messaggi solo da questi server. Pertanto, una determinata richiesta `ReceiveMessage` potrebbe non restituire tutti i messaggi. Tuttavia, se hai meno di 1.000 messaggi in coda, una richiesta successiva restituirà i tuoi messaggi. Se continui a utilizzare messaggi dalle code, Amazon SQS campiona tutti i propri server e riceverai tutti i messaggi.

Il diagramma qui di seguito mostra il comportamento di short-polling dei messaggi restituiti da una coda standard dopo che uno dei componenti del sistema invia una richiesta di ricezione. Amazon SQS campiona diversi server (in grigio) e restituisce i messaggi A, C, D e B da questi server. Il messaggio E non viene restituito in questa richiesta, ma viene restituito in una richiesta successiva.



Utilizzo di messaggi con long polling

Quando il tempo di attesa per l'azione dell'API `ReceiveMessage` è superiore a 0, viene attivato un polling lungo. Il tempo massimo di attesa per il polling lungo è di 20 secondi. Il long polling aiuta a ridurre il costo di utilizzo di Amazon SQS eliminando il numero di risposte vuote (quando non ci sono messaggi disponibili per una richiesta `ReceiveMessage`) e le risposte vuote false (quando i messaggi sono disponibili, ma non sono inclusi nella risposta). Per informazioni su come abilitare il polling lungo per una coda nuova o esistente utilizzando la console Amazon SQS, consulta Configurazione dei parametri della coda (console). Per le best practice, consulta Impostazione di polling lungo.

Il long polling offre i seguenti benefici:

- Elimina le risposte vuote consentendo a Amazon SQS di attendere fino a che un messaggio non è disponibile nella coda prima di inviare una risposta. A meno che la connessione non scada, la risposta alla richiesta `ReceiveMessage` contiene almeno uno dei messaggi disponibili, fino al numero massimo di messaggi specificato nell'operazione `ReceiveMessage`. In rari casi, potresti ricevere risposte vuote anche quando una coda contiene ancora messaggi, specialmente se hai specificato un valore basso per il parametro `ReceiveMessageWaitTimeSeconds`.
- Riduci le false risposte vuote interrogando tutti i server Amazon SQS, anziché un sottoinsieme di essi.
- Restituisci i messaggi non appena diventano disponibili.

Per informazioni su come confermare che una coda è vuota, consulta Verifica che una coda sia vuota.

Differenze tra short e long polling

Lo short polling si verifica quando il parametro `WaitTimeSeconds` di una richiesta `ReceiveMessage` è impostato su `0` in uno dei seguenti due modi:

- La chiamata `ReceiveMessage` imposta `WaitTimeSeconds` su `0`.
- La chiamata `ReceiveMessage` non imposta `WaitTimeSeconds` e l'attributo coda `ReceiveMessageWaitTimeSeconds` è impostato su `0`.

Code DLQ di Amazon SQS

Amazon SQS supporta le code DLQ a cui altre code (code sorgenti) possono inviare i messaggi che non vengono elaborati correttamente (consumati). Le code DLQ sono utili per il debug di un'applicazione o di un sistema di messaggistica perché consentono di isolare messaggi non consumati e stabilire il motivo per cui la loro elaborazione non è riuscita. Per informazioni su come configurare code DLQ utilizzando la console Amazon SQS, consulta Configurazione di una coda DLQ (console). Una volta eseguito il debug dell'applicazione consumer o dopo che l'applicazione consumer è disponibile a consumare il messaggio, puoi utilizzare la funzionalità di redrive della coda DLQ per spostare i messaggi alla coda di origine.

> **Important**
>
> Amazon SQS non crea automaticamente la coda DLQ. Occorre innanzitutto creare un coda prima di designarla come coda DLQ.

Argomenti

- Come funzionano le code DLQ?
- Quali sono i vantaggi delle code DLQ?
- In che modo tipi di code diversi gestiscono gli errori dei messaggi?
- Quando devo usare una coda DLQ?
- Spostare i messaggi fuori da una coda DLQ
- Risoluzione dei problemi relativi alle code DLQ

- Configurazione di una coda DLQ (console)
- Configurazione di un redrive della coda DLQ
- Requisiti di aggiornamento e autorizzazione di CloudTrail per il redrive della coda DLQ di Amazon SQS

Come funzionano le code DLQ?

A volte, i messaggi non possono essere elaborati a causa di un'ampia gamma di problemi possibile, ad esempio condizioni di errore all'interno dell'applicazione del produttore o consumatore o una modifica imprevista dello stato che determina un problema con il codice dell'applicazione. Ad esempio, se un utente inserisce un ordine Web con un determinato ID prodotto, ma l'ID prodotto viene eliminato, il codice del web store ha esito negativo e viene visualizzato un messaggio di errore e il messaggio con la richiesta dell'ordine viene inviato a una coda DLQ.

Occasionalmente, produttori e consumatori potrebbero non riuscire a interpretare aspetti del protocollo che utilizzano per comunicare, causando la perdita o il danneggiamento del messaggio. Inoltre, gli errori hardware del consumatore possono danneggiare il payload del messaggio.

La policy di reindirizzamento specifica la coda di origine, la coda DLQ e le condizioni in cui Amazon SQS sposta i messaggi dalla prima alla seconda se il consumatore della coda di origine non riesce a elaborare un messaggio per un determinato numero di volte. `maxReceiveCount` è il numero di volte in cui un consumatore tenta di ricevere un messaggio da una coda senza eliminarlo prima di essere spostato nella coda DLQ. Se si imposta `maxReceiveCount` su un valore basso, ad esempio 1, la mancata ricezione del messaggio comporterebbe lo spostamento del messaggio nella coda DLQ. Tali errori includono errori di rete ed errori di dipendenza del client. Per garantire la resilienza del sistema contro gli errori, imposta un valore `maxReceiveCount` sufficientemente alto da consentire un numero sufficiente di tentativi.

La policy redrive allow specifica quali code di origine possono accedere a una coda DLQ. Questa policy si applica a una potenziale coda DLQ. È possibile scegliere se consentire tutte le code di origine, consentire code di origine specifiche o negare tutte le code di origine. L'impostazione predefinita prevede che tutte le code di origine utilizzino la coda DLQ. Se scegli di consentire code specifiche (utilizzando l'opzione `byQueue`), puoi specificare fino a 10 code di origine utilizzando la coda di origine Amazon Resource Name (ARN). Se si specifica `denyAll`, la coda non può essere utilizzata come coda DLQ.

Per specificare una coda DLQ puoi utilizzare la console o gli SDK AWS. Devi eseguire questa operazione per ogni coda che invia messaggi a una coda DLQ. Code multiple dello stesso tipo

possono scegliere come target una singola coda di lettere morte. Per ulteriori informazioni, consulta Configurazione di una coda DLQ (console) e gli attributi `RedrivePolicy` e `RedriveAllowPolicy` dell'operazione `CreateQueue` o `SetQueueAttributes`.

> **Important**
>
> La coda DLQ di una coda FIFO deve anche essere una coda FIFO. Analogamente, la coda DLQ di una coda standard deve anche essere una coda standard.
> Devi utilizzare lo stesso Account AWS per creare la coda DLQ e le altre code che inviano messaggi alla coda DLQ. Inoltre, le code DLQ devono trovarsi nella stessa regione in cui si trovano le altre code che utilizzano la coda DLQ. Ad esempio, se crei una coda nella regione Stati Uniti orientali (Ohio) e desideri utilizzare una coda DLQ con tale coda, anche la seconda coda deve trovarsi nella regione Stati Uniti orientali (Ohio).
> Per le code standard, la scadenza di un messaggio si basa sempre sul timestamp della coda originale. Quando un messaggio viene spostato in una coda DLQ, il timestamp della coda rimane invariato. La metrica `ApproximateAgeOfOldestMessage` indica quando il messaggio è stato spostato nella coda DLQ, non quando è stato originariamente inviato. Ad esempio, supponiamo che un messaggio trascorra 1 giorno nella coda originale prima di essere spostato in una coda DLQ. Se il periodo di conservazione della coda DLQ è di 4 giorni, il messaggio viene eliminato dalla coda DLQ dopo 3 giorni e `ApproximateAgeOfOldestMessage`è di tre giorni. Pertanto, è consigliabile impostare sempre il periodo di conservazione di una coda DLQ in modo che sia più lungo del periodo di conservazione della coda originale.
> Per le code FIFO, il timestamp della coda si reimposta quando il messaggio viene spostato in una coda DLQ. La metrica `ApproximateAgeOfOldestMessage` indica quando il messaggio è stato spostato nella coda DLQ. Nello stesso esempio precedente, il messaggio viene eliminato dalla coda DLQ dopo 4 giorni e `ApproximateAgeOfOldestMessage` è 4 giorni.

Quali sono i vantaggi delle code DLQ?

Il compito principale di una coda DLQ è gestire il ciclo di vita dei messaggi non consumati. Una coda DLQ ti consente di riservare e isolare i messaggi che non possono essere elaborati correttamente per stabilire il motivo per cui l'elaborazione non è riuscita. La configurazione di una coda DLQ consente di eseguire le operazioni seguenti:

- Configurare un allarme per qualsiasi messaggio inviato a una coda DLQ.
- Esaminare i log per trovare eccezioni che potrebbero aver causato il trasferimento dei messaggi a una coda DLQ.
- Analizzare i contenuti dei messaggi recapitati a una coda DLQ per diagnosticare problemi software o problemi hardware del produttore o consumatore.
- Determinare se hai concesso al tuo consumatore tempo sufficiente per elaborare i messaggi.

In che modo tipi di code diversi gestiscono gli errori dei messaggi?

Code standard

Le code standard mantengono l'elaborazione dei messaggi fino alla scadenza del periodo di conservazione. Ciò garantisce l'elaborazione continua dei messaggi, che riduce al minimo la possibilità che la coda venga bloccata da messaggi che non possono essere elaborati. L'elaborazione continua dei messaggi consente inoltre un ripristino più rapido della coda.

In un sistema che elabora migliaia di messaggi, disporre di un numero elevato di messaggi che il consumatore ripetutamente non riesce a riconoscere ed eliminare potrebbe far aumentare i costi e collocare un maggiore carico sull'hardware. Anziché tentare di elaborare i messaggi con esito negativo finché non scadono, è preferibile spostarli in una coda DLQ dopo pochi tentativi di elaborazione.

> **Note**
>
> Le code standard consentono un elevato numero di messaggi in transito. Se la maggior parte dei messaggi non può essere utilizzata e i messaggi non vengono inviati a una coda DLQ; la velocità di elaborazione dei messaggi validi può diminuire. Pertanto, per mantenere l'efficienza della coda, devi accertarti che l'applicazione gestisca l'elaborazione del messaggio correttamente.

Code FIFO

Le code FIFO garantiscono l'elaborazione "exactly-once" utilizzando messaggi in sequenza da un gruppo di messaggi. Pertanto, anche se il consumatore può continuare a recuperare messaggi ordinati da un altro gruppo di messaggi, il primo non è disponibile finché il messaggio che blocca la coda non viene elaborato correttamente o spostato a una coda DLQ.

> **Note**
>
> Le code FIFO consentono un numero inferiore di messaggi in transito. Pertanto, per garantire che la tua coda FIFO non venga bloccata da un messaggio, devi accertarti che la tua applicazione gestisca correttamente l'elaborazione del messaggio.
> Quando un messaggio viene spostato da una coda FIFO a una DLQ FIFO, l'ID di deduplicazione del messaggio originale verrà sostituito con l'ID del messaggio originale. Ciò serve a garantire che la deduplicazione DLQ non impedisca l'archiviazione di due messaggi indipendenti che condividono un ID di deduplicazione.

Quando devo usare una coda DLQ?

✔ Utilizza le code DLQ con code standard. È sempre consigliabile sfruttare le code DLQ quando le applicazioni non fanno affidamento sull'ordinamento. Le code DLQ possono aiutarti a risolvere i problemi relativi a operazioni di trasmissione messaggi errate.

> **Note**
>
> Anche quando utilizzi code DLQ, è consigliabile continuare a monitorare le code e riprovare a inviare i messaggi che hanno avuto esito negativo per motivi transitori.

✔ Utilizza le code DLQ per ridurre il numero di messaggi e la possibilità di esporre il sistema a messaggi poison-pill (messaggi che possono essere ricevuti ma non elaborati).

✘ Non utilizzare una coda DLQ con code standard quando desideri continuare a ritentare la trasmissione di un messaggio a tempo indeterminato. Ad esempio, non utilizzare una coda DLQ se il tuo programma deve attendere che un processo dipendente diventi attivo o disponibile.

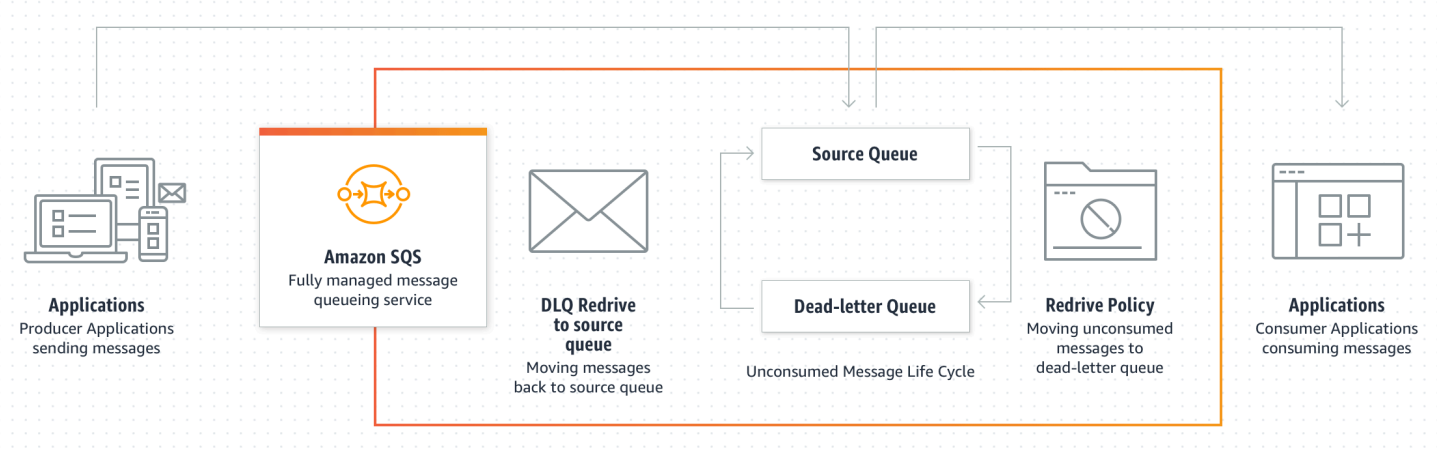
✘ Non utilizzare una coda DLQ con una coda FIFO se non desideri interrompere l'ordine esatto di messaggi oppure operazioni. Ad esempio, non utilizzare una coda DLQ con istruzioni in un EDL (Edit Decision

List) per una suite di editing video, dove la modifica dell'ordine delle modifiche modifica il contesto delle modifiche successive.

Spostare i messaggi fuori da una coda DLQ

È possibile utilizzare il redrive della coda DLQ per gestire il ciclo di vita dei messaggi non utilizzati. Dopo aver esaminato gli attributi e i relativi metadati disponibili per i messaggi non utilizzati in una coda standard o DLQ FIFO, è possibile reindirizzare i messaggi alle code di origine. Il redrive della coda DLQ riduce la fatturazione delle chiamate API raggruppando i messaggi in batch durante lo spostamento.

L'attività di redrive utilizza le API `SendMessageBatch`, `ReceiveMessage` e `DeleteMessageBatch` di Amazon SQS per conto dell'utente per reindirizzare i messaggi. Pertanto, tutti i messaggi reindirizzati sono considerati nuovi messaggi con nuovi `messageid`, `enqueueTime` e periodo di conservazione. Il prezzo del redrive della coda DLQ utilizza il numero di chiamate API richiamate e le fatture in base ai prezzi di Amazon SQS.



Per impostazione predefinita, il redrive della coda DLQ sposta i messaggi da una coda DLQ a una coda di origine. Tuttavia, puoi anche configurare qualsiasi altra coda come destinazione di redrive se entrambe le code sono di tipo uguale. Ad esempio, se la coda DLQ è una coda FIFO, anche la coda di destinazione di redrive deve essere una coda FIFO. Inoltre, puoi configurare la velocità di redrive per impostare la velocità con cui Amazon SQS sposta i messaggi. Per istruzioni sulla configurazione di un redrive per una coda DLQ, consulta Configurazione di un redrive della coda DLQ.

> **Note**
>
> Amazon SQS non supporta il filtraggio e la modifica dei messaggi mentre li reindirizza dalla coda DLQ.

Un'attività di redrive di una coda DLQ può essere eseguita per un massimo di 36 ore. Amazon SQS supporta un massimo di 100 attività di redrive attive per account.
Quando i messaggi vengono reindirizzati da una coda DLQ FIFO, il groupID e il deduplicationID del messaggio rimangono gli stessi e il messaggio riceve un nuovo messageID.
Le code DLQ di Amazon SQS reindirizzano i messaggi nell'ordine in cui vengono ricevuti, a partire dal messaggio più vecchio. Tuttavia, la coda di destinazione inserisce i messaggi reindirizzati, così come i nuovi messaggi degli altri produttori, in base all'ordine in cui li riceve. Ad esempio, se un produttore invia messaggi a una coda FIFO di origine e riceve contemporaneamente messaggi reindirizzati da una coda DLQ, i messaggi reindirizzati si mescoleranno ai nuovi messaggi del produttore.

Risoluzione dei problemi relativi alle code DLQ

In alcuni casi, le code DLQ di Amazon SQS non si comportano come previsto. Questa sezione fornisce una panoramica dei problemi più comuni e illustra come risolverli.

La visualizzazione dei messaggi utilizzando la console può comportarne il trasferimento a una coda DLQ

Amazon SQS conteggia la visualizzazione di un messaggio nella console rispetto alle policy di redrive della coda corrispondente. Pertanto, se visualizzi un messaggio nella console il numero di volte specificato nella policy di reindirizzamento della coda corrispondente, il messaggio viene spostato nella coda DLQ corrispondente.

Per correggere questo comportamento, puoi procedere in uno dei seguenti modi:

- Aumentare l'impostazione Maximum Receives (Ricezioni massime) per la policy di reindirizzamento della coda corrispondente.
- Evitare di visualizzare i messaggi della coda corrispondente nella console.

`NumberOfMessagesSent` e **`NumberOfMessagesReceived`** per una coda DLQ non corrispondono

Se invii un messaggio a una coda DLQ manualmente, viene acquisito dal parametro `NumberOfMessagesSent`. Tuttavia, se viene inviato un messaggio a una coda DLQ a seguito di un

tentativo di elaborazione non riuscito, non viene acquisito da questo parametro. Pertanto, è possibile che i valori di `NumberOfMessagesSent` e `NumberOfMessagesReceived` siano differenti.

Per informazioni sulla creazione e la configurazione di un redrive della coda DLQ

Tieni presente che il redrive della coda DLQ richiede l'impostazione delle autorizzazioni appropriate per Amazon SQS per ricevere messaggi dalla coda DLQ e inviare messaggi alla coda di destinazione. In caso di autorizzazioni insufficienti, il redrive della coda DLQ alla coda di origine non avvia il redrive dei messaggi e l'operazione può fallire. È possibile visualizzare lo stato dell'attività di redrive dei messaggi per risolvere i problemi e riprovare.

Argomenti

- Configurazione di una coda DLQ (console)
- Configurazione di un redrive della coda DLQ
- Requisiti di aggiornamento e autorizzazione di CloudTrail per il redrive della coda DLQ di Amazon SQS

Configurazione di una coda DLQ (console)

Scopri come configurare una coda DLQ che una o più code di origine possono utilizzare per i messaggi che non vengono consumati correttamente. Per ulteriori informazioni, consulta Code DLQ di Amazon SQS.

Amazon SQS non crea automaticamente la coda DLQ. Occorre innanzitutto creare un coda prima di designarla come coda DLQ. Per istruzioni sulla creazione di una coda da utilizzare come coda DLQ, consulta Creazione di una coda (console)

La coda DLQ di una coda FIFO deve anche essere una coda FIFO. Analogamente, la coda DLQ di una coda standard deve anche essere una coda standard.

Quando si crea o si modifica una coda, è possibile configurare una coda DLQ.

Configurazione di una coda DLQ per una coda esistente (console).

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code).
3. Scegli una coda e seleziona Modifica.

4. Scorri fino alla sezione Coda DLQ e scegli Abilitato.
5. Scegli il nome della risorsa Amazon (ARN) di una coda DLQ esistente che desideri associare a questa coda di origine.
6. Per configurare il numero di volte che un messaggio può essere ricevuto prima di essere inviato a una coda DLQ, imposta Ricezioni massime su un valore compreso tra 1 e 1.000.
7. Al termine della configurazione della coda DLQ, scegli Salva.

   Dopo aver salvato la coda, la console visualizza la pagina Dettagli per la coda. Nella pagina Dettagli, la scheda Coda DLQ visualizza l'ARN Maximum Receives e Coda DLQ nella coda Dead-Letter.

Configurazione di un redrive della coda DLQ

È possibile configurare un redrive della coda DLQ per spostare i messaggi standard non utilizzati da una coda DLQ esistente alle relative code di origine. Per ulteriori informazioni sul redrive della coda DLQ, consulta Spostare i messaggi fuori da una coda DLQ.

Configurazione di un redrive della coda DLQ per una coda standard (API) esistente

Puoi configurare un redrive della coda DLQ utilizzando le seguenti operazioni API.

Azione API	Descrizione
`StartMessageMoveTask`	Avvia un'attività asincrona per spostare i messaggi da una coda di origine specificata a una coda di destinazione specificata.
`ListMessageMoveTasks`	Ottiene le attività di spostamento dei messaggi più recenti (fino a 10) in una coda di origine specifica.
`CancelMessageMoveTask`	Annulla un'operazione di spostamento dei messaggi specificata. Lo spostamento di un messaggio può essere annullato solo quando lo stato corrente è IN ESECUZIONE.

Configurazione di un redrive della coda DLQ per una coda standard (console) esistente

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code).
3. Scegli il nome della coda che hai configurato come coda DLQ.
4. Scegli Avvia redrive DLQ.
5. In Configurazione Redrive, per Destinazione del messaggio, esegui una delle seguenti operazioni:
   - Per reindirizzare i messaggi nella relativa coda di origine, scegli Reindirizza verso le code di origine.
   - Per reindirizzare i messaggi su un'altra coda, scegli Reindirizza verso una destinazione personalizzata. Quindi, immettere il nome della risorsa Amazon (ARN) di una coda di destinazione esistente.

     > **Note**
     >
     > La coda di destinazione personalizzata deve corrispondere al tipo della coda DLQ. Ad esempio, se la coda DLQ è una coda FIFO, anche la coda di destinazione personalizzata deve essere una coda FIFO.

6. In Impostazioni per la regolazione della velocità, scegli una delle seguenti opzioni:
   - Ottimizzato per il sistema: reindirizza i messaggi della coda DLQ al numero massimo di messaggi al secondo.
   - Velocità massima personalizzata: reindirizza i messaggi in coda DLQ con una frequenza massima personalizzata di messaggi al secondo. La frequenza massima consentita è di 500 messaggi al secondo.
     - Si consiglia di iniziare con un valore basso per la velocità massima personalizzata e verificare che la coda di origine non sia sovraccarica di messaggi. Da lì, aumentare gradualmente il valore di velocità massima personalizzata, continuando a monitorare lo stato della coda di origine.
7. Al termine della configurazione del redrive della coda DLQ, scegli Messaggi di redrive.

> **Important**
>
> Amazon SQS non supporta il filtraggio e la modifica dei messaggi mentre li reindirizza dalla coda DLQ.
> Un'attività di redrive di una coda DLQ può essere eseguita per un massimo di 36 ore.
> Amazon SQS supporta un massimo di 100 attività di redrive attive per account.
> L'attività di redrive reimposta il periodo di conservazione. Nuovi `messageID` e `enqueueTime` vengono assegnati ai messaggi reindirizzati.

8. Se desideri annullare l'operazione di redrive dei messaggi, nella pagina Dettagli della coda, scegli Annulla redrive DLQ. Quando si annulla un redrive di messaggio in corso, tutti i messaggi che sono già stati spostati correttamente nella coda di destinazione dello spostamento rimarranno nella coda di destinazione.

Configurazione delle autorizzazioni per il redrive della coda DLQ

Puoi concedere agli utenti l'accesso a specifiche operazioni relative alla coda DLQ aggiungendo autorizzazioni alla tua policy. Le autorizzazioni minime richieste per reindirizzare una coda DLQ sono le seguenti:

Autorizzazioni minime	Metodi API richiesti
Per avviare il reindirizzamento di un messaggio	• Aggiungi `sqs:StartMessageMoveTask` , `sqs:ReceiveMessage` , `sqs:DeleteMessage` e `sqs:GetQueueAttributes` della coda DLQ. Se la coda DLQ o la coda dell'origine sono crittografate (nota anche come coda SSE), è necessaria anche `kms:Decrypt` per la chiave KMS utilizzata per crittografare i messaggi. • Aggiungere `sqs:SendMessage` della coda di destinazione. Se la coda di destinazione è crittografata, sono inoltre obbligatori `kms:GenerateDataKey` e `kms:Decrypt` .
Per annullare il redrive di un	• Aggiungere `sqs:CancelMessageMoveTask` , `sqs:ReceiveMessage` , `sqs:DeleteMessage` e `sqs:GetQueueAttrib`

Autorizzazioni minime	Metodi API richiesti
messaggio in corso	`utes` della coda DLQ. Se la coda DLQ è crittografata (nota anche come coda SSE) è necessario anche `kms:Decrypt` .
Per mostrare lo stato di spostamento di un messaggio	• Aggiungere `sqs:ListMessageMoveTasks` e `sqs:GetQueueAttributes` della coda DLQ.

Per configurare le autorizzazioni per una coppia di coda crittografata (una coda di origine con una coda DLQ)

Utilizza i seguenti passaggi per configurare le autorizzazioni minime per il redrive di una coda DLQ:

1. Accedi a AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione, seleziona Policies (Policy).
3. Crea una policy con le seguenti autorizzazioni e collegala al tuo utente o ruolo IAM:
   - `sqs:StartMessageMoveTask`
   - `sqs:CancelMessageMoveTask`
   - `sqs:ListMessageMoveTasks`
   - `sqs:ListDeadLetterSourceQueues`
   - `sqs:ReceiveMessage`
   - `sqs:DeleteMessage`
   - `sqs:GetQueueAttributes`
   - L'ARN `Resource` della coda DLQ (for example, "arn:aws:sqs:*<DLQ_region>*:*<DLQ_accountId>*:*<DLQ_name>*").
   - `sqs:SendMessage`
   - L'`Resource`ARN della coda di destinazione (ad esempio, «arn:aws:sqs: *< DestQueue _region>: < _accountID>: < _name>* «). DestQueue DestQueue
   - `kms:Decrypt`: consente l'azione di decrittografia.
   - `kms:GenerateDataKey`

- Gli ARN `Resource` di qualsiasi chiave di crittografia KMS utilizzata per crittografare i messaggi nella coda di origine originale (ad esempio, "arn:aws:kms:*<region>*:*<accountId>*:key/*<keyId_used to encrypt the message body>*").
- L'ARN della risorsa della chiave di crittografia KMS utilizzata per la coda di destinazione del redrive (ad esempio, "arn:aws:kms:*<region>*:*<accountId>*:key/*<keyId_used for the destination queue>*").

La policy di accesso deve essere simile alla seguente:

```
{
"Version": "2012-10-17",
  "Statement": [
    {
"Effect": "Allow",
      "Action": [
        "sqs:StartMessageMoveTask",
        "sqs:CancelMessageMoveTask",
        "sqs:ListMessageMoveTasks",
        "sqs:ReceiveMessage",
        "sqs:DeleteMessage",
        "sqs:GetQueueAttributes"
      ],
      "Resource": "arn:aws:sqs:<DLQ_region>:<DLQ_accountId>:<DLQ_name>"
    },
    {
"Effect": "Allow",
      "Action": "sqs:SendMessage",
      "Resource":
 "arn:aws:sqs:<DestQueue_region>:<DestQueue_accountId>:<DestQueue_name>"
    },
    {
"Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
      ],
      "Resource": "arn:aws:kms:<region>:<accountId>:key/<keyId>"
    }
  ]
```

```
}
```

Per configurare le autorizzazioni per una coppia di coda non crittografata (una coda di origine con una coda DLQ)

Utilizza i seguenti passaggi per configurare le autorizzazioni minime per una coda DLQ standard non-crittografata. Le autorizzazioni minime richieste sono per ricevere, eliminare e recuperare gli attributi dalla coda DLQ e inviare attributi alla coda di origine.

1. Accedi a AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione, seleziona Policies (Policy).
3. Crea una policy con le seguenti autorizzazioni e collegala al tuo utente o ruolo IAM:

   - `sqs:StartMessageMoveTask`
   - `sqs:CancelMessageMoveTask`
   - `sqs:ListMessageMoveTasks`
   - `sqs:ReceiveMessage`
   - `sqs:DeleteMessage`
   - `sqs:GetQueueAttributes`
   - L'ARN `Resource` della coda DLQ (ad esempio, "arn:aws:sqs:*<DLQ_region>*:*<DLQ_accountId>*:*<DLQ_name>*").
   - `sqs:SendMessage`
   - *L'ResourceARN della coda di destinazione (ad esempio, «arn:aws:sqs: < DestQueue _region>: < _accountID>: < _name> «). DestQueue DestQueue*

   La policy di accesso deve essere simile alla seguente:

```
{
"Version": "2012-10-17",
  "Statement": [
    {
"Effect": "Allow",
      "Action": [
        "sqs:StartMessageMoveTask",
        "sqs:CancelMessageMoveTask",
```

```
        "sqs:ListMessageMoveTasks",
        "sqs:ReceiveMessage",
        "sqs:DeleteMessage",
        "sqs:GetQueueAttributes"
      ],
      "Resource": "arn:aws:sqs:<DLQ_region>:<DLQ_accountId>:<DLQ_name>"
    },
    {
"Effect": "Allow",
      "Action": "sqs:SendMessage",
      "Resource":
 "arn:aws:sqs:<DestQueue_region>:<DestQueue_accountId>:<DestQueue_name>"
    }
  ]
}
```

Requisiti di aggiornamento e autorizzazione di CloudTrail per il redrive della coda DLQ di Amazon SQS

L'8 giugno 2023, Amazon SQS ha introdotto il redrive DLQ (dead-letter queue) per AWS SDK e AWS Command Line Interface (CLI). Questa funzionalità è un'aggiunta al redrive DLQ già supportato per la console AWS. Se in precedenza hai utilizzato la console AWS per reindirizzare i messaggi della coda DLQ, potrebbero interessarti le seguenti modifiche:

- Rinomina degli eventi CloudTrail per il redrive della coda DLQ
- Autorizzazioni aggiornate per il redrive della coda DLQ

Ridenominazione di eventi CloudTrail

Il 15 ottobre 2023, i nomi degli eventi CloudTrail per il redrive delle code DLQ cambieranno sulla console Amazon SQS. Se hai impostato allarmi per questi eventi CloudTrail, devi aggiornarli ora. Di seguito sono riportati i nuovi nomi degli eventi CloudTrail per il redrive DLQ:

Nome evento precedente	Nuovo nome evento
`CreateMoveTask`	`StartMessageMoveTask`
`CancelMoveTask`	`CancelMessageMoveTask`

Autorizzazioni aggiornate

Incluso nella versione SDK e CLI, Amazon SQS ha anche aggiornato le autorizzazioni di coda per il redrive DLQ per aderire alle best practice di sicurezza. Utilizza i seguenti tipi di autorizzazione alla coda per reindirizzare i messaggi dalle tue DLQ.

1. Autorizzazioni basate sulle azioni (aggiornamento per le azioni dell'API DLQ)
2. Autorizzazioni della policy di Amazon SQS gestita
3. Policy di autorizzazione che utilizza caratteri jolly sqs:*

> **Important**
>
> Per utilizzare il redrive DLQ per SDK o CLI, è necessario disporre di una policy di autorizzazione di redrive DLQ che corrisponda a una delle opzioni precedenti.

Se le autorizzazioni di coda per il redrive DLQ non corrispondono a una delle opzioni precedenti, devi aggiornare le autorizzazioni entro il 31 agosto 2023. Entro il 31 agosto 2023, il tuo account sarà in grado di reindirizzare i messaggi utilizzando le autorizzazioni configurate tramite la console AWS solo nelle regioni in cui hai precedentemente utilizzato il redrive DLQ. Ad esempio, supponiamo di avere un “Account A" sia in us-east-1 che in eu-west-1. L'"Account A" è stato utilizzato per reindirizzare i messaggi sulla console AWS in us-east-1 prima dell'8 giugno 2023, ma non in eu-west-1. Tra l'8 giugno 2023 e il 31 agosto 2023, se le autorizzazioni della policy "Account A" non corrispondono a una delle opzioni precedenti, può essere utilizzato solo per reindirizzare i messaggi sulla console AWS in us-east-1 e non in eu-west-1.

> **Important**
>
> Se le tue autorizzazioni di redrive DLQ non corrispondono a una di queste opzioni dopo il 31 agosto 2023, il tuo account non sarà più in grado di reindirizzare i messaggi DLQ utilizzando la console AWS.
>
> Tuttavia, se hai utilizzato la funzione di redrive DLQ sulla console AWS nel mese di agosto 2023, hai un'estensione fino al 15 ottobre 2023 per adottare le nuove autorizzazioni in base a una di queste opzioni.
>
> Per ulteriori informazioni, consulta the section called “Identificazione delle policy interessate”.

Di seguito sono riportati alcuni esempi di autorizzazioni di coda per ogni opzione di redrive DLQ. Quando si utilizzano code crittografate sul lato server (SSE), è richiesta l'autorizzazione della chiave AWS KMS corrispondente.

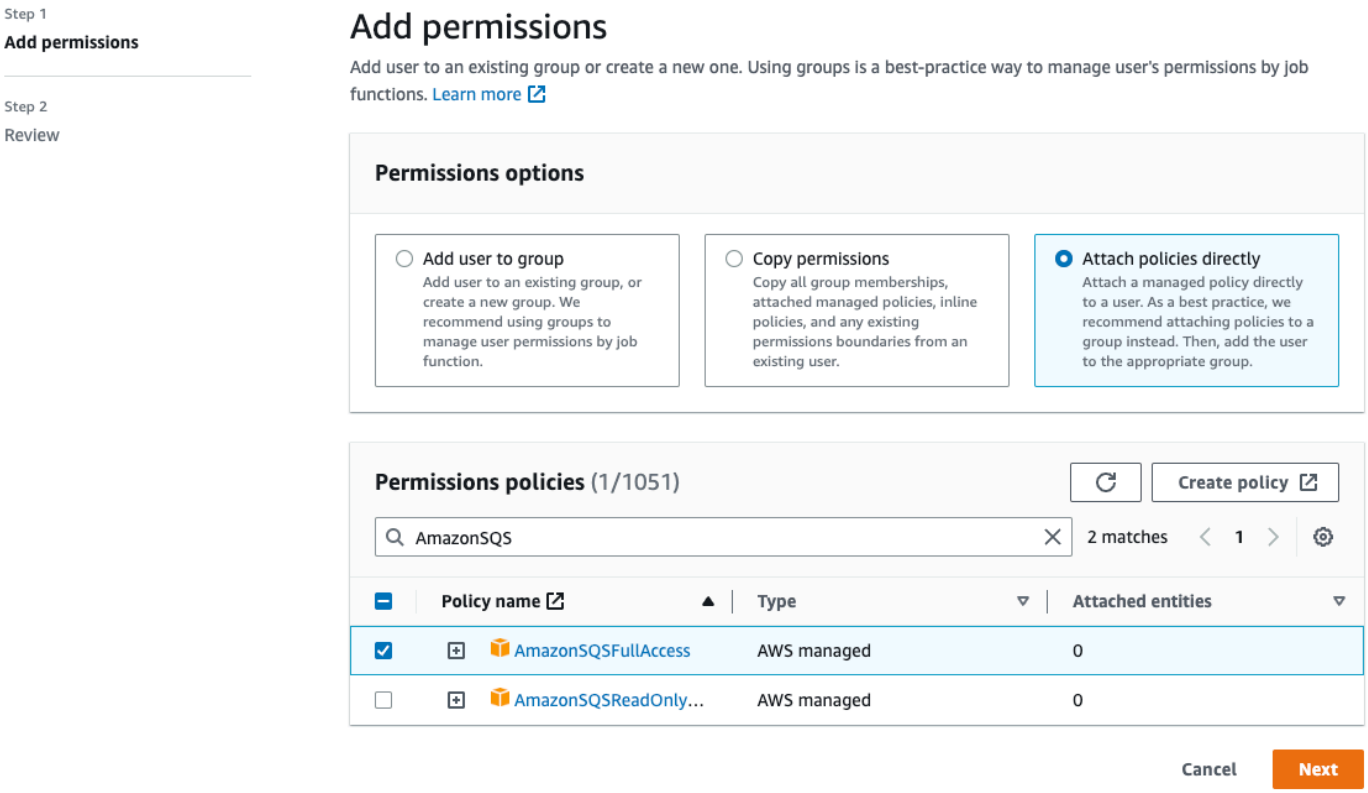
Basato sull'azione

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sqs:ReceiveMessage",
        "sqs:DeleteMessage",
        "sqs:GetQueueAttributes",
        "sqs:StartMessageMoveTask",
        "sqs:ListMessageMoveTasks",
        "sqs:CancelMessageMoveTask"
      ],
      "Resource": "arn:aws:sqs:<DLQ_region>:<DLQ_accountId>:<DLQ_name>"
    },
    {
      "Effect": "Allow",
      "Action": "sqs:SendMessage",
      "Resource":
 "arn:aws:sqs:<DestQueue_region>:<DestQueue_accountId>:<DestQueue_name>"
    }
  ]
}
```

Policy gestita

Le seguenti policy gestite contengono le autorizzazioni aggiornate richieste:

- AmazonSQSFullAccess: include le seguenti attività di reindirizzamento delle code DLQ: avvio, annullamento ed elenco.
- AmazonSQSReadOnlyAccess: fornisce accesso in sola lettura e include l'attività di redrive della coda DLQ.



Policy di autorizzazione che utilizza caratteri jolly sqs*

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sqs:*",
      "Resource": "*"
    }
  ]
}
```

Identificazione delle policy interessate

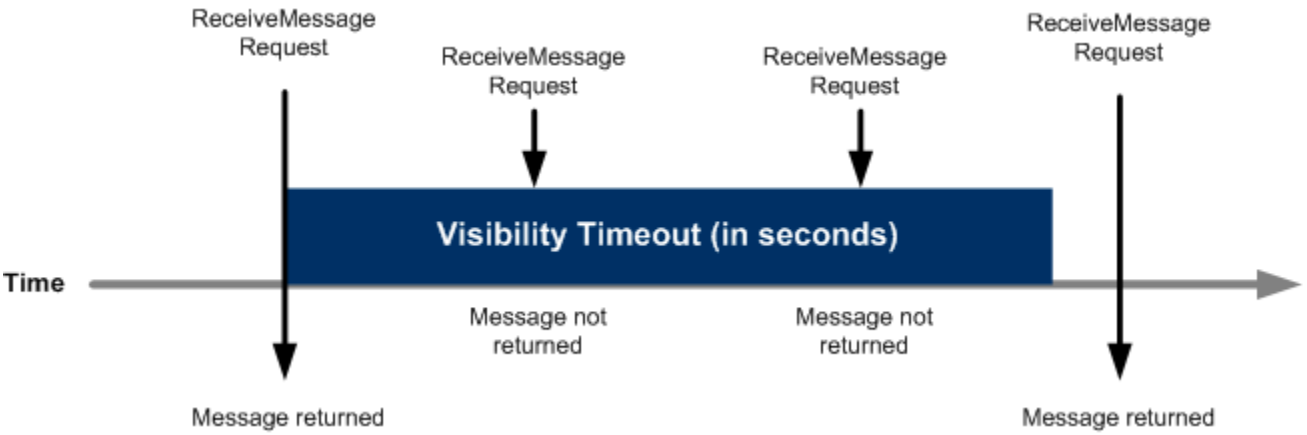
Se si utilizzano policy gestite dai clienti (CMP), è possibile utilizzare AWS CloudTrail e IAM per identificare le politiche interessate dall'aggiornamento delle autorizzazioni di coda.

> **Note**
>
> Se si utilizza `AmazonSQSFullAccess` e `AmazonSQSReadOnlyAccess`, non sono necessarie ulteriori azioni.

1. Accedere alla console AWS CloudTrail.
2. Nella pagina Cronologia degli eventi, in Cerca attributi, utilizza il menu a discesa per selezionare Nome evento. Quindi, cerca `CreateMoveTask`.
3. Scegli un evento per aprire la pagina dei Dettagli. Nella sezione Record degli eventi, recuperare `UserName` o `RoleName` dall'ARN `userIdentity`.
4. Accedi alla console IAM.
   - Per gli utenti, scegli Utenti. Selezionare l'utente con i dati `UserName` identificati nella fase precedente.
   - Per i ruoli, scegli Ruoli. Selezionare l'utente con i dati `RoleName` identificati nella fase precedente.
5. Nella pagina Dettagli, nella sezione Autorizzazioni, esamina tutte le politiche con il prefisso `sqs:` prefisso in `Action` o esamina le politiche in cui è definita la coda Amazon SQS in `Resource`.

Timeout visibilità Amazon SQS

Quando un consumatore riceve ed elabora un messaggio da una coda, il messaggio rimane nella coda. Amazon SQS non elimina automaticamente il messaggio. Poiché Amzon SQS è un sistema distribuito, non vi è alcuna garanzia che il consumatore effettivamente riceva il messaggio (ad esempio, a causa di un problema di connessione, oppure a causa di un problema nell'applicazione del consumatore). Di conseguenza, il consumatore deve eliminare il messaggio dalla coda dopo la ricezione e l'elaborazione.



Immediatamente dopo che il messaggio viene ricevuto, rimane nella coda. Per prevenire che altri consumatori elaborino nuovamente il messaggio, Amazon SQS imposta un timeout visibilità, un periodo di tempo durante il quale Amazon SQS impedisce ad altri consumatori di ricevere ed elaborare il messaggio. Il timeout visibilità predefinito per una coda è di 30 secondi. Il valore minimo è 0 secondi. La durata massima è 12 ore. Per ulteriori informazioni sulla configurazione del timeout di visibilità per una coda mediante la console, consultare Configurazione dei parametri della coda (console).

Note

Per le code standard il timeout visibilità non garantisce che un messaggio non venga ricevuto due volte. Per ulteriori informazioni, consulta Una consegna t-least-once .
Le code FIFO consentono al produttore o al consumatore di fare più tentativi:

- Se il produttore rileva un'azione `SendMessage` non riuscita, può ritentare l'invio tutte le volte che è necessario, utilizzando lo stesso ID di deduplicazione del messaggio. Supponendo che il produttore riceva almeno una conferma prima della scadenza dell'intervallo di deduplicazione, più tentativi non influiscono sull'ordine dei messaggi né introducono duplicati.
- Se il consumatore rileva un'azione `ReceiveMessage` non riuscita, può riprovare tutte le volte che è necessario, utilizzando lo stesso ID del tentativo di richiesta di ricezione. Supponendo che il consumatore riceva almeno una conferma prima della scadenza del timeout di visibilità, più tentativi non influiscono sull'ordine dei messaggi.
- Quando ricevi un messaggio con un ID gruppo di messaggi, non vengono restituiti altri messaggi per lo stesso ID gruppo di messaggi a meno che non elimini il messaggio o non diventi visibile.

Argomenti

Messaggi in transito

Un messaggio Amazon SQS ha tre stati di base:

1. Inviato a una coda da un produttore.
2. Ricevuto dalla coda da un consumatore.
3. Eliminato dalla coda.

Un messaggio viene considerato archiviato dopo essere stato inviato a una coda da un produttore, ma non ancora ricevuto dalla coda da un consumatore (ossia, tra gli stati 1 e 2). Non è prevista alcuna quota per il numero di messaggi archiviati. Un messaggio viene considerato in transito dopo essere stato ricevuto da una coda da un consumatore, ma non ancora eliminato dalla coda (ossia, tra gli stati 2 e 3). Non è prevista alcuna quota per il numero di messaggi in transito.

> **Important**
>
> Le quote che si applicano ai messaggi in transito non sono correlate al numero illimitato di messaggi archiviati.

Per la maggior parte delle code standard (a seconda del traffico in coda e del backlog di messaggi), possono esserci un massimo di circa 120.000 messaggi in transito (ricevuti da una coda da un consumatore, ma non ancora eliminati dalla coda). Se si raggiunge questa quota durante l'utilizzo del polling breve, Amazon SQS restituisce il messaggio di errore `OverLimit`. Se si utilizza un long polling, Amazon SQS non restituisce alcun messaggio di errore. Per evitare di raggiungere la quota, elimina i messaggi dalla coda dopo che sono stati elaborati. Puoi anche aumentare il numero di code utilizzate per elaborare i messaggi. Per richiedere un incremento della quota, invia una richiesta di supporto.

Per le code FIFO, possono esserci un massimo di 20.000 messaggi in corso (ricevuti da una coda da un consumatore, ma non ancora eliminati dalla coda). Se raggiungi questa quota, Amazon SQS non restituisce alcun messaggio di errore.

> **Important**
>
> Quando si lavora con le code FIFO, le operazioni `DeleteMessage` falliranno se la richiesta viene ricevuta al di fuori della finestra di timeout di visibilità. Se il timeout di visibilità è 0 secondi, il messaggio deve essere eliminato entro lo stesso millisecondo in cui è stato inviato, altrimenti viene considerato abbandonato. Ciò può far sì che Amazon SQS includa messaggi duplicati nella stessa risposta a un'operazione `ReceiveMessage` se il parametro `MaxNumberOfMessages` è maggiore di 1. Per ulteriori dettagli, consulta Come funziona l'API FIFO di Amazon SQS.

Impostazione del timeout visibilità

Il timeout visibilità inizia quando Amazon SQS restituisce un messaggio. Durante tale periodo, il consumatore elabora ed elimina il messaggio. Tuttavia, se il consumatore ha esito negativo prima di eliminare il messaggio e il tuo sistema non chiama l'operazione `DeleteMessage` per quel messaggio prima della scadenza di un timeout visibilità, il messaggio diventa visibile ad altri consumatori e viene ricevuto nuovamente. Se un messaggio deve essere ricevuto solo una volta, il tuo consumatore deve eliminarlo entro la durata del timeout visibilità.

Per ogni coda Amazon SQS, l'impostazione del timeout visibilità predefinita è di 30 secondi. Puoi modificare questa impostazione per l'intera coda. Di solito devi impostare il timeout visibilità sul tempo massimo necessario alla tua applicazione per elaborare ed eliminare un messaggio dalla coda. Quando ricevi messaggi, puoi anche impostare un timeout visibilità speciali per i messaggi restituiti senza modificare il timeout della coda generale. Per ulteriori informazioni, consulta le best practice nella sezione Elaborazione di messaggi in modo tempestivo.

Se non sai quanto tempo sia necessario per elaborare un messaggio, crea un heartbeat per il processo del consumatore: specifica il timeout visibilità iniziale (ad esempio, 2 minuti) e poi, se il consumatore lavora ancora sul messaggio, continua a estendere il timeout visibilità di 2 minuti ogni minuto.

> **Important**
>
> Il timeout massimo di visibilità è di 12 ore dal momento in cui Amazon SQS riceve la richiesta `ReceiveMessage`. L'estensione del timeout di visibilità non reimposta il massimo di 12 ore. Inoltre, potresti non essere in grado di impostare il timeout per un singolo messaggio per tutte le 12 ore (ad esempio 43.200 secondi) trascorse dalla richiesta `ReceiveMessage` che avvia il timer. Ad esempio, se si riceve un messaggio e si imposta immediatamente il limite massimo di 12 ore inviando una chiamata `ChangeMessageVisibility` della durata di 43.200 secondi, è probabile che la chiamata abbia esito negativo. Tuttavia, l'utilizzo di un valore di 43.195 secondi funzionerà a meno che non vi sia un ritardo significativo tra la richiesta del messaggio tramite `ReceiveMessage` e l'aggiornamento del timeout di visibilità. Se il consumatore ha bisogno di più di 12 ore, prendere in considerazione l'utilizzo di Step Functions.

Modifica del timeout visibilità per un messaggio

Quando ricevi un messaggio da una coda e inizi a elaborarlo, il timeout visibilità per la coda può essere insufficiente (ad esempio, potresti dover elaborare ed eliminare un messaggio). Puoi accorciare o estendere la visibilità di un messaggio specificando un nuovo valore di timeout utilizzando l'operazione `ChangeMessageVisibility`.

Ad esempio, se il timeout predefinito per una coda è di 60 secondi, sono trascorsi 15 secondi da quando è stato ricevuto il messaggio e invii una chiamata `ChangeMessageVisibility` con `VisibilityTimeout` impostato su 10 secondi, i 10 secondi iniziano a essere contati dal momento in cui effettui la chiamata `ChangeMessageVisibility`. Pertanto, qualsiasi tentativo di modificare il timeout visibilità o di eliminare il messaggio 10 secondi dopo aver inizialmente modificato il timeout visibilità (un totale di 25 secondi) può generare un errore.

> **Note**
>
> Il nuovo periodo di timeout diventa effettivo dal momento in cui si chiama l'operazione `ChangeMessageVisibility`. Inoltre, il nuovo periodo di timeout si applica solo alla particolare ricezione del messaggio. `ChangeMessageVisibility` non influenza il timeout di ricezioni successive del messaggio o di code successive.

Interruzione del timeout visibilità per un messaggio

Quando ricevi un messaggio da una coda, potresti capire che effettivamente non vuoi elaborarlo ed eliminarlo. Amazon SQS ti consente di terminare il timeout visibilità per un determinato messaggio. In questo modo il messaggio diventa immediatamente visibile ad altri componenti nel sistema e disponibile per l'elaborazione.

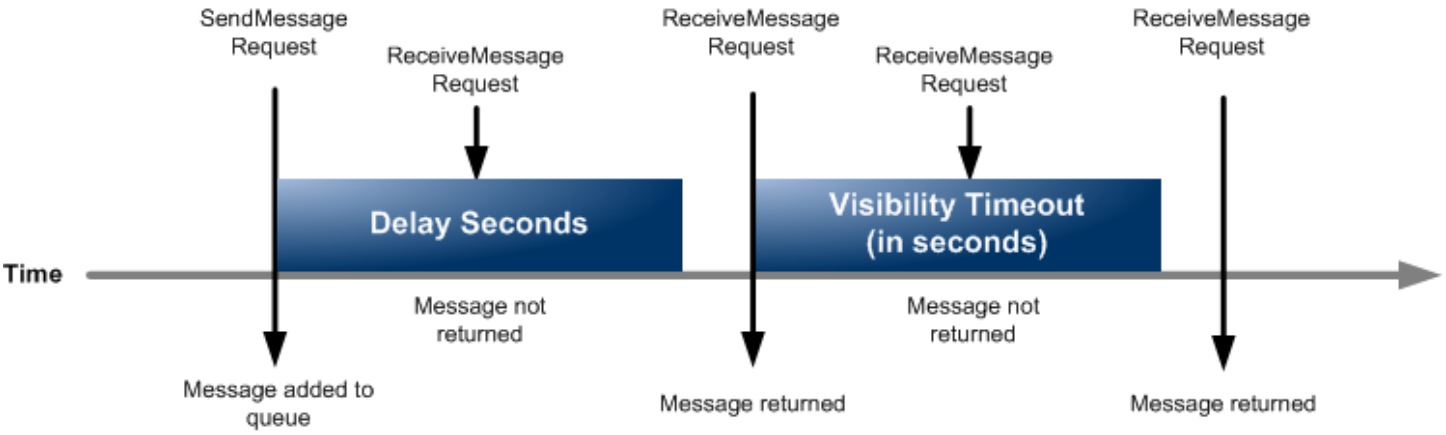
Per terminare il timeout visibilità di un messaggio dopo aver chiamato `ReceiveMessage`, chiama `ChangeMessageVisibility` con `VisibilityTimeout` impostato su 0 secondi.

Code di ritardo Amazon SQS

Le code di ritardo consentono di posticipare la consegna di nuovi messaggi ai consumatori per alcuni secondi, ad esempio quando l'applicazione consumer richiede più tempo per elaborare i messaggi. Se crei una coda di ritardo, qualsiasi messaggio inviato a tale coda rimane invisibile ai consumatori per la durata del periodo di ritardo. Il ritardo predefinito (minimo) per una coda è 0 secondi. Il valore massimo è 15 minuti. Per ulteriori informazioni sulla configurazione delle code di ritardo mediante la console, consultare Configurazione dei parametri della coda (console).

> **Note**
>
> Per le code standard, l'impostazione di ritardo per coda non è retroattiva: se modifichi l'impostazione, non influisce sul ritardo dei messaggi già nella coda.
> Per le code FIFO, l'impostazione di ritardo per coda è retroattiva: se modifichi l'impostazione, influisce sul ritardo dei messaggi già nella coda.

Le code di ritardo sono simili al timeout visibilità perché entrambe le caratteristiche rendono non disponibili i messaggi ai consumatori per un determinato periodo di tempo. La differenza tra le code di ritardo e i timeout visibilità è che per le code di ritardo un messaggio viene nascosto quando viene aggiunto per la prima volta alla coda, mentre per i timeout visibilità un messaggio viene nascosto solo dopo che un messaggio viene consumato dalla coda. Il seguente diagramma mostra il rapporto tra le code di ritardo e i timeout visibilità.



Per impostare i secondi di ritardo in singoli messaggi, anziché su un'intera coda, utilizza i timer messaggio per consentire ad Amazon SQS di utilizzare il valore `DelaySeconds` del timer messaggio, anziché il valore `DelaySeconds` della coda del ritardo.

Code temporanee di Amazon SQS

Le code temporanee consentono di risparmiare tempo di sviluppo e costi di implementazione quando si utilizzano schemi di messaggi comuni come request-response. È possibile utilizzare il Temporary Queue Client per creare code temporanee ad alta velocità, convenienti e gestite dalle applicazioni.

Il client mappa automaticamente più code temporanee, code gestite da applicazioni create su richiesta per un particolare processo, su una singola coda Amazon SQS. In questo modo l'applicazione può effettuare meno chiamate API e incrementare il throughput quando il traffico per ogni coda temporanea è basso. Quando una coda temporanea non è più in uso, il client la pulisce automaticamente, anche se alcuni processi che utilizzano il client non sono chiusi correttamente.

Di seguito sono elencati i vantaggi delle code temporanee:

- Agiscono da canali di comunicazione leggeri per specifici thread o processi.
- Possono essere creati ed eliminati senza incorrere in costi aggiuntivi.
- Sono compatibili a livello di API con le code statiche (normali) Amazon SQS. Ciò significa che il codice esistente che invia e riceve messaggi può inviare e ricevere messaggi dalle code virtuali.

Argomenti

Code virtuali

Le code virtuali sono strutture di dati locali create da Temporary Queue Client. Le code virtuali consentono di combinare più destinazioni a traffico ridotto in un'unica coda Amazon SQS. Per le best practice, consulta Evitare di riutilizzare lo stesso ID gruppo di messaggi con le code virtuali.

> **Note**
>
> - Quando si crea una coda virtuale vengono create solo le strutture di dati temporanee in cui i consumatori ricevono i messaggi. Dato che una coda virtuale non effettua chiamate API a Amazon SQS, le code virtuali non prevedono costi.
> - Le quote TPS si applicano a tutte le code virtuali in una singola coda host. Per ulteriori informazioni, consulta Quote correlate ai messaggi.

La classe wrapper `AmazonSQSVirtualQueuesClient` aggiunge il supporto per gli attributi relativi alle code virtuali. Per creare una coda virtuale, è necessario chiamare l'operazione API `CreateQueue` utilizzando l'attributo `HostQueueURL`. Questo attributo specifica la coda esistente che ospita le code virtuali.

L'URL di una coda virtuale è nel formato seguente.

```
https://sqs.us-east-2.amazonaws.com/123456789012/MyQueue#MyVirtualQueueName
```

Quando un produttore chiama l'operazione API `SendMessageBatch` o `SendMessage` sull'URL di una coda virtuale, Temporary Queue Client esegue queste operazioni:

1. Estrae il nome della coda virtuale.
2. Collega il nome della coda virtuale come ulteriore attributo del messaggio.
3. Invia il messaggio alla coda host.

Mentre il produttore invia messaggi, un thread in background esegue il polling della coda host e invia i messaggi ricevuti alle code virtuali in base ai corrispondenti attributi di messaggio.

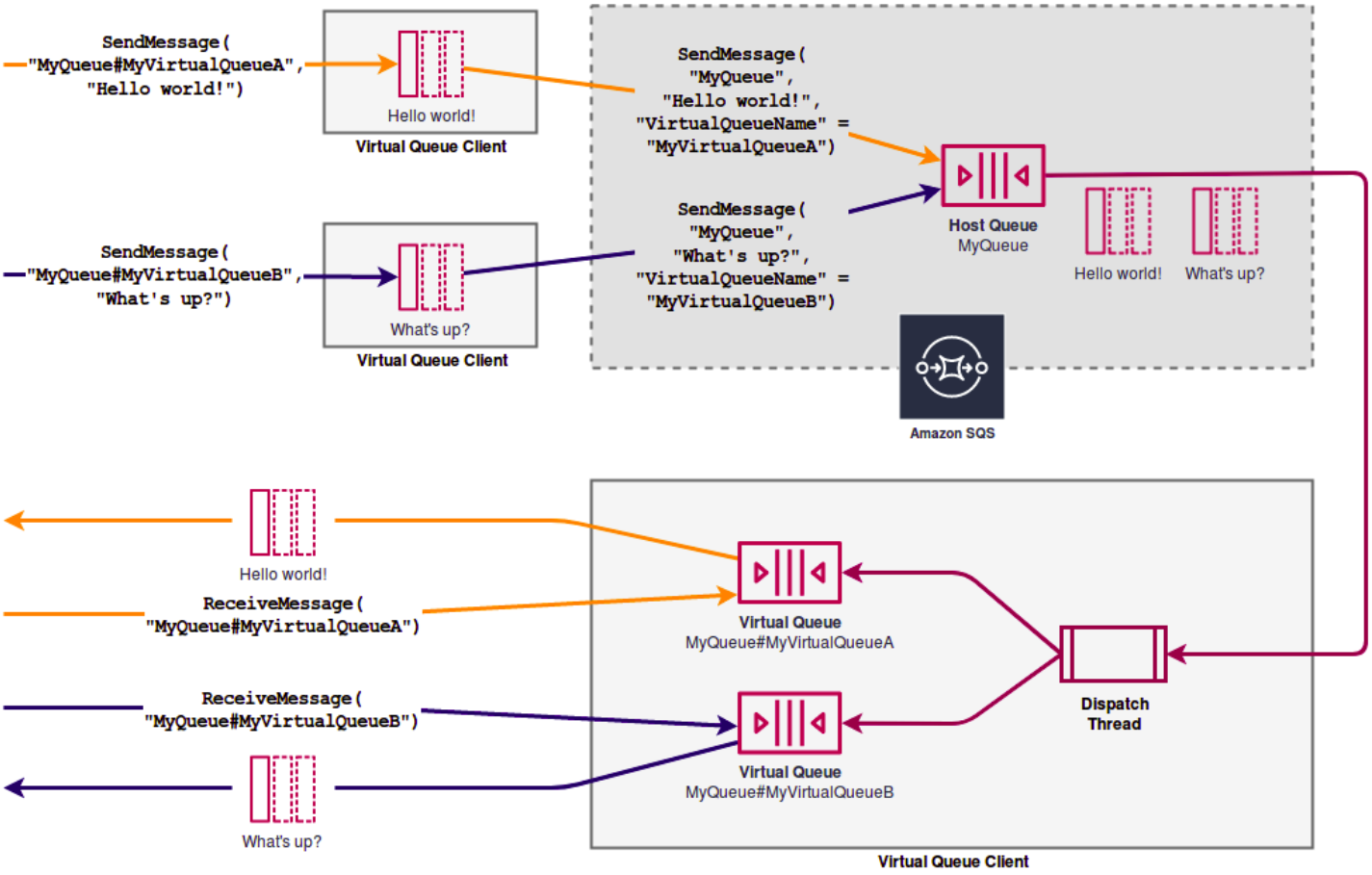
Mentre il consumatore chiama l'operazione API `ReceiveMessage` sull'URL di una coda virtuale, Temporary Queue Client blocca la chiamata in locale finché il thread in background non invia un messaggio alla coda virtuale. Si tratta di una procedura simile al prefetching del messaggio nel Buffered Asynchronous Client: una singola operazione API può fornire messaggi fino a 10 code virtuali). L'eliminazione di una coda virtuale rimuove tutte le risorse lato client, senza chiamare Amazon SQS.

La classe `AmazonSQSTemporaryQueuesClient` trasforma automaticamente tutte le code che crea in code temporanee. Inoltre, crea automaticamente le code host con gli stessi attributi della coda, su richiesta. Questi nomi di code condividono un prefisso comune configurabile (per impostazione predefinita `__RequesterClientQueues__`) che li identifica come code temporanee. In questo modo il client può agire da sostituzione drop-in che ottimizza la coda esistente che crea ed elimina le code. Il client include anche le interfacce `AmazonSQSRequester` e `AmazonSQSResponder` che consentono la comunicazione bidirezionale tra le code.

Modelli di messaggistica richiesta-risposta (code virtuali)

Il caso d'uso più comune per le code temporanee è il modello di messaggistica richiesta-risposta, in cui un richiedente crea una coda temporanea per la ricezione di ciascun messaggio di risposta. Per evitare di creare una coda Amazon SQS per ogni messaggio di risposta, Temporary Queue Client consente di creare ed eliminare più code temporanee senza effettuare chiamate API Amazon SQS. Per ulteriori informazioni, consulta Implementazione di sistemi Richiesta-Risposta.

Il diagramma seguente mostra una configurazione comune che usa questo modello.



Scenario di esempio: elaborazione di una richiesta di accesso

Il seguente scenario di esempio mostra come usare le interfacce `AmazonSQSResponder` e `AmazonSQSRequester` per elaborare la richiesta di accesso di un utente.

Sul lato client

```
public class LoginClient {

    // Specify the Amazon SQS queue to which to send requests.
    private final String requestQueueUrl;

    // Use the AmazonSQSRequester interface to create
    // a temporary queue for each response.
    private final AmazonSQSRequester sqsRequester =
            AmazonSQSRequesterClientBuilder.defaultClient();

    LoginClient(String requestQueueUrl) {
        this.requestQueueUrl = requestQueueUrl;
```

```
    }

    // Send a login request.
    public String login(String body) throws TimeoutException {
        SendMessageRequest request = new SendMessageRequest()
                .withMessageBody(body)
                .withQueueUrl(requestQueueUrl);

        // If no response is received, in 20 seconds,
        // trigger the TimeoutException.
        Message reply = sqsRequester.sendMessageAndGetResponse(request,
                20, TimeUnit.SECONDS);

        return reply.getBody();
    }
}
```

L'invio di una richiesta di accesso esegue queste operazioni:

1. Crea una coda temporanea.
2. Collega l'URL della coda temporanea al messaggio come attributo.
3. Invia il messaggio.
4. Riceve una risposta dalla coda temporanea.
5. Elimina la coda temporanea.
6. Restituisce la risposta.

Sul lato server

L'esempio seguente presuppone che, al momento della creazione, venga creato un thread per eseguire il polling della coda e chiamare il metodo `handleLoginRequest()` per ogni messaggio. Inoltre, presuppone l'uso del metodo `doLogin()`.

```
public class LoginServer {

    // Specify the Amazon SQS queue to poll for login requests.
    private final String requestQueueUrl;

    // Use the AmazonSQSResponder interface to take care
    // of sending responses to the correct response destination.
    private final AmazonSQSResponder sqsResponder =
```

```
            AmazonSQSResponderClientBuilder.defaultClient();

    LoginServer(String requestQueueUrl) {
        this.requestQueueUrl = requestQueueUrl;
    }

    // Process login requests from the client.
    public void handleLoginRequest(Message message) {

        // Process the login and return a serialized result.
        String response = doLogin(message.getBody());

        // Extract the URL of the temporary queue from the message attribute
        // and send the response to the temporary queue.
        sqsResponder.sendResponseMessage(MessageContent.fromMessage(message),
                new MessageContent(response));
    }
}
```

Pulizia delle query

Per garantire che Amazon SQS recuperi tutte le risorse in memoria utilizzate dalle code virtuali, quando l'applicazione non ha più bisogno di Temporary Queue Client, deve chiamare il metodo `shutdown()`. Puoi anche utilizzare il metodo `shutdown()` dell'interfaccia `AmazonSQSRequester`.

Temporary Queue Client, inoltre, fornisce un modo per eliminare le code host orfane. Per ogni coda che riceve una chiamata API in un periodo di tempo (per impostazione predefinita, cinque minuti), il client utilizza l'operazione API `TagQueue` per applicare tag a una coda che rimane in uso.

> **Note**
>
> Qualsiasi operazione API eseguita su una coda la contrassegna come non inattiva, compresa un'operazione `ReceiveMessage` che non restituisce messaggi.

Il thread in background usa le operazioni API `ListQueues` e `ListTags` per controllare tutte le code con il prefisso configurato, eliminando quelle che non sono state contrassegnate con tag per almeno cinque minuti. In questo modo, se un client non si chiude correttamente, gli altri client attivi vengono puliti dopo di esso. Per ridurre le duplicazioni di lavoro, tutti i client con lo stesso prefisso comunicano attraverso una coda di lavoro interna condivisa, il cui nome corrisponde al prefisso.

Timer di messaggi Amazon SQS

I timer messaggio consentono di specificare un periodo di invisibilità iniziale per un messaggio che viene aggiunto a una coda. Ad esempio, se invii un messaggio con il timer impostato su 45 secondi, il messaggio non è visibile ai consumatori per i primi 45 secondi durante il quale il messaggio rimane nella coda. Il ritardo predefinito (minimo) per un messaggio è 0 secondi. Il valore massimo è 15 minuti. Per ulteriori informazioni sull'invio di messaggi con timer tramite la console, consulta Invio di un messaggio.

> **Note**
>
> Le code FIFO non supportano i timer sui singoli messaggi.

Per impostare un periodo di ritardo su un'intera coda, piuttosto che su singoli messaggi, utilizza le code di ritardo. Un'impostazione di timer messaggio per un singolo messaggio sostituisce il valore `DelaySeconds` che si applica a una coda di ritardo Amazon SQS.

Accesso ad Amazon EventBridge Pipes tramite la console Amazon SQS

Amazon EventBridge Pipes collega le sorgenti alle destinazioni. Le pipe sono destinate point-to-point alle integrazioni tra sorgenti e destinazioni supportate, con supporto per trasformazioni e arricchimenti avanzati. EventBridge Le pipe offrono un modo altamente scalabile per connettere la coda Amazon SQS AWS a servizi come Step Functions, Amazon SQS e API Gateway, nonché ad applicazioni SaaS (Software as a Service) di terze parti come Salesforce.

Per configurare una pipe, si sceglie l'origine, si aggiungono filtri facoltativi, si definisce l'arricchimento facoltativo e si sceglie la destinazione per i dati dell'evento.

Nella pagina dei dettagli di una coda Amazon SQS, puoi visualizzare le pipe che utilizzano quella coda come origine. Da lì, puoi anche:

- Avvia la console per visualizzare i dettagli delle tubazioni. EventBridge
- Avvia la EventBridge console per creare una nuova pipe con la coda come sorgente.

Per ulteriori informazioni sulla configurazione di una coda Amazon SQS come sorgente pipe, consulta Amazon SQS queue as a source nella Amazon User Guide. EventBridge Per ulteriori informazioni sulle EventBridge pipe in generale, consulta Pipes. EventBridge

Per accedere alle EventBridge pipe per una determinata coda Amazon SQS

1. Nella console Amazon SQS, apri la pagina delle code.
2. Seleziona una coda.
3. Nella pagina dei dettagli della coda, scegli la scheda Pipes. EventBridge

   La scheda EventBridge Pipes include un elenco di tutte le pipe attualmente configurate per utilizzare la coda selezionata come sorgente, tra cui:

   - nome della pipe
   - stato corrente
   - destinazione della pipe
   - quando la pipe è stata modificata l'ultima volta
4. Visualizza altri dettagli sulla pipe o crea una nuova pipe, se lo desideri:

   - Accesso a dettagli aggiuntivi su una pipe:

     Scegliere il nome della pipe.

     Verrà avviata la pagina dei dettagli di Pipe della EventBridge console.
   - Creazione di una nuova pipe:

     Scegli Connetti la coda Amazon SQS alla pipe.

     Questo avvia la pagina Create pipe della EventBridge console, con la coda Amazon SQS specificata come origine della pipe. Per ulteriori informazioni, consulta Creating an EventBridge pipe nella Amazon EventBridge User Guide.

     > **Important**
     >
     > Un messaggio su una coda Amazon SQS viene letto da una singola pipe e quindi eliminato dalla coda dopo l'elaborazione, indipendentemente dal fatto che il messaggio corrisponda o meno al filtro configurato per quella pipe. Quando configuri più pipe fai attenzione in modo che utilizzino la stessa coda come sorgente.

Gestione di messaggi Amazon SQS di grandi dimensioni con Extended Client Library e Amazon Simple Storage Service

Puoi utilizzare Amazon SQS Extended Client Library per Java e Amazon SQS Extended Client Library for Python per inviare messaggi di grandi dimensioni. Ciò è particolarmente utile per consumare carichi utili di messaggi di grandi dimensioni, da 256 KB a 2 GB. Entrambe le librerie salvano il payload del messaggio in un bucket Amazon Simple Storage Service e inviano il riferimento dell'oggetto Amazon S3 archiviato alla coda Amazon SQS.

> **Note**
>
> Le librerie Amazon SQS Extended Client sono compatibili con le code Standard e FIFO.

Argomenti

Gestione di messaggi Amazon SQS di grandi dimensioni con Java e Amazon S3

Puoi quindi utilizzare Amazon SQS Extended Client Library per Java e Amazon Simple Storage Service (Amazon S3) per gestire messaggi Amazon Simple Queue Service (Amazon SQS) di grandi dimensioni. Ciò è particolarmente utile per consumare carichi di messaggi di grandi dimensioni, da 256 KB a 2 GB. La libreria salva il payload del messaggio in un bucket Amazon S3 e invia un messaggio contenente un riferimento dell'oggetto Amazon S3 archiviato a una coda Amazon Amazon SQS.

Puoi utilizzare la libreria client ampia di Amazon SQS per Java per completare le seguenti operazioni:

- Specificare se i messaggi vengono sempre archiviati in Amazon S3 o solo quando le dimensioni di un messaggio superano i 256 KB.
- Inviare un messaggio che fa riferimento a un solo oggetto messaggio archiviato in un bucket S3
- Recuperare l'oggetto messaggio da un bucket S3
- Eliminare l'oggetto messaggio da un bucket S3

Prerequisiti

Gli esempi seguenti utilizzano AWS Java SDK. Per installare e configurare l'SDK, consulta Impostare il AWS SDK per Java nella AWS SDK for JavaGuida per gli sviluppatori.

Prima di eseguire il codice di esempio, configurare le credenziali AWS. Per ulteriori informazioni, consulta Configurazione delle credenziali e della regione AWS per lo sviluppo nella Guida per gli sviluppatori di AWS SDK for Java.

SDK per Java e la libreria client ampia Amazon SQS per Java richiedono J2SE Development Kit 8.0 o versioni successive.

> **Note**
>
> Puoi utilizzare la libreria client ampia Amazon SQS per Java per gestire i messaggi Amazon SQS utilizzando Amazon S3 solo con AWS SDK for Java. Non puoi farlo con AWS CLI, la console Amazon SQS, l'API HTTP di Amazon SQS o qualsiasi altro SDK AWS.

AWSSDK for Java 1.x Esempio: utilizzo di Amazon S3 per gestire messaggi Amazon SQS di grandi dimensioni

Il seguente esempio di AWSSDK for Java 2.x crea un bucket Amazon S3 con un nome casuale e aggiunge una regola del ciclo di vita per eliminare definitivamente gli oggetti dopo 14 giorni. Inoltre, crea una coda denominata `MyQueue` e invia un messaggio casuale che viene archiviato in un bucket S3 e la coda supera i 256 KB. Infine, il codice utilizza il messaggio, restituisce informazioni su di esso e poi elimina il messaggio, la coda e il bucket.

```
/*
 * Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
```

```
 *
 */

import com.amazon.sqs.javamessaging.AmazonSQSExtendedClient;
import com.amazon.sqs.javamessaging.ExtendedClientConfiguration;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;
import com.amazonaws.services.sqs.AmazonSQS;
import com.amazonaws.services.sqs.AmazonSQSClientBuilder;
import com.amazonaws.services.sqs.model.*;
import org.joda.time.DateTime;
import org.joda.time.format.DateTimeFormat;

import java.util.Arrays;
import java.util.List;
import java.util.UUID;

public class SQSExtendedClientExample {

// Create an Amazon S3 bucket with a random name.
private final static String S3_BUCKET_NAME = UUID.randomUUID() + "-"
        + DateTimeFormat.forPattern("yyMMdd-hhmmss").print(new DateTime());

public static void main(String[] args) {

    /*
     * Create a new instance of the builder with all defaults (credentials
     * and region) set automatically. For more information, see
     * Creating Service Clients in the AWS SDK for Java Developer Guide.
     */
    final AmazonS3 s3 = AmazonS3ClientBuilder.defaultClient();

    /*
     * Set the Amazon S3 bucket name, and then set a lifecycle rule on the
     * bucket to permanently delete objects 14 days after each object's
     * creation date.
     */
    final BucketLifecycleConfiguration.Rule expirationRule =
            new BucketLifecycleConfiguration.Rule();
    expirationRule.withExpirationInDays(14).withStatus("Enabled");
    final BucketLifecycleConfiguration lifecycleConfig =
            new BucketLifecycleConfiguration().withRules(expirationRule);
```

```
    // Create the bucket and allow message objects to be stored in the bucket.
    s3.createBucket(S3_BUCKET_NAME);
    s3.setBucketLifecycleConfiguration(S3_BUCKET_NAME, lifecycleConfig);
    System.out.println("Bucket created and configured.");

    /*
     * Set the Amazon SQS extended client configuration with large payload
     * support enabled.
     */
    final ExtendedClientConfiguration extendedClientConfig =
            new ExtendedClientConfiguration()
                    .withLargePayloadSupportEnabled(s3, S3_BUCKET_NAME);

    final AmazonSQS sqsExtended =
            new AmazonSQSExtendedClient(AmazonSQSClientBuilder
                    .defaultClient(), extendedClientConfig);

    /*
     * Create a long string of characters for the message object which will
     * be stored in the bucket.
     */
    int stringLength = 300000;
    char[] chars = new char[stringLength];
    Arrays.fill(chars, 'x');
    final String myLongString = new String(chars);

    // Create a message queue for this example.
    final String QueueName = "MyQueue" + UUID.randomUUID().toString();
    final CreateQueueRequest createQueueRequest =
            new CreateQueueRequest(QueueName);
    final String myQueueUrl = sqsExtended
            .createQueue(createQueueRequest).getQueueUrl();
    System.out.println("Queue created.");

    // Send the message.
    final SendMessageRequest myMessageRequest =
            new SendMessageRequest(myQueueUrl, myLongString);
    sqsExtended.sendMessage(myMessageRequest);
    System.out.println("Sent the message.");

    // Receive the message.
    final ReceiveMessageRequest receiveMessageRequest =
            new ReceiveMessageRequest(myQueueUrl);
    List<Message> messages = sqsExtended
```

```
            .receiveMessage(receiveMessageRequest).getMessages();

    // Print information about the message.
    for (Message message : messages) {
        System.out.println("\nMessage received.");
        System.out.println("  ID: " + message.getMessageId());
        System.out.println("  Receipt handle: " + message.getReceiptHandle());
        System.out.println("  Message body (first 5 characters): "
                + message.getBody().substring(0, 5));
    }

    // Delete the message, the queue, and the bucket.
    final String messageReceiptHandle = messages.get(0).getReceiptHandle();
    sqsExtended.deleteMessage(new DeleteMessageRequest(myQueueUrl,
            messageReceiptHandle));
    System.out.println("Deleted the message.");

    sqsExtended.deleteQueue(new DeleteQueueRequest(myQueueUrl));
    System.out.println("Deleted the queue.");

    deleteBucketAndAllContents(s3);
    System.out.println("Deleted the bucket.");
}

private static void deleteBucketAndAllContents(AmazonS3 client) {

    ObjectListing objectListing = client.listObjects(S3_BUCKET_NAME);

    while (true) {
        for (S3ObjectSummary objectSummary : objectListing
                .getObjectSummaries()) {
            client.deleteObject(S3_BUCKET_NAME, objectSummary.getKey());
        }

        if (objectListing.isTruncated()) {
            objectListing = client.listNextBatchOfObjects(objectListing);
        } else {
            break;
        }
    }

    final VersionListing list = client.listVersions(
            new ListVersionsRequest().withBucketName(S3_BUCKET_NAME));
```

```
    for (S3VersionSummary s : list.getVersionSummaries()) {
        client.deleteVersion(S3_BUCKET_NAME, s.getKey(), s.getVersionId());
    }

    client.deleteBucket(S3_BUCKET_NAME);
}
}
```

AWSSDK for Java 2.x Esempio: utilizzo di Amazon S3 per gestire messaggi Amazon SQS di grandi dimensioni

Il seguente esempio di AWSSDK for Java 2.x crea un bucket Amazon S3 con un nome casuale e aggiunge una regola del ciclo di vita per eliminare definitivamente gli oggetti dopo 14 giorni. Inoltre, crea una coda denominata `MyQueue` e invia un messaggio casuale che viene archiviato in un bucket S3 e la coda supera i 256 KB. Infine, il codice utilizza il messaggio, restituisce informazioni su di esso e poi elimina il messaggio, la coda e il bucket.

```
/*
 * Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

            import com.amazon.sqs.javamessaging.AmazonSQSExtendedClient;
import com.amazon.sqs.javamessaging.ExtendedClientConfiguration;
import org.joda.time.DateTime;
import org.joda.time.format.DateTimeFormat;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.BucketLifecycleConfiguration;
import software.amazon.awssdk.services.s3.model.CreateBucketRequest;
import software.amazon.awssdk.services.s3.model.DeleteBucketRequest;
import software.amazon.awssdk.services.s3.model.DeleteObjectRequest;
```

```
import software.amazon.awssdk.services.s3.model.ExpirationStatus;
import software.amazon.awssdk.services.s3.model.LifecycleExpiration;
import software.amazon.awssdk.services.s3.model.LifecycleRule;
import software.amazon.awssdk.services.s3.model.LifecycleRuleFilter;
import software.amazon.awssdk.services.s3.model.ListObjectVersionsRequest;
import software.amazon.awssdk.services.s3.model.ListObjectVersionsResponse;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Response;
import software.amazon.awssdk.services.s3.model.PutBucketLifecycleConfigurationRequest;
import software.amazon.awssdk.services.sqs.SqsClient;
import software.amazon.awssdk.services.sqs.model.CreateQueueRequest;
import software.amazon.awssdk.services.sqs.model.CreateQueueResponse;
import software.amazon.awssdk.services.sqs.model.DeleteMessageRequest;
import software.amazon.awssdk.services.sqs.model.DeleteQueueRequest;
import software.amazon.awssdk.services.sqs.model.Message;
import software.amazon.awssdk.services.sqs.model.ReceiveMessageRequest;
import software.amazon.awssdk.services.sqs.model.ReceiveMessageResponse;
import software.amazon.awssdk.services.sqs.model.SendMessageRequest;

import java.util.Arrays;
import java.util.List;
import java.util.UUID;


/**
 * Examples of using Amazon SQS Extended Client Library for Java 2.x
 *
 */
public class SqsExtendedClientExamples {
    // Create an Amazon S3 bucket with a random name.
    private final static String S3_BUCKET_NAME = UUID.randomUUID() + "-"
            + DateTimeFormat.forPattern("yyMMdd-hhmmss").print(new DateTime());

    public static void main(String[] args) {

        /*
         * Create a new instance of the builder with all defaults (credentials
         * and region) set automatically. For more information, see
         * Creating Service Clients in the AWS SDK for Java Developer Guide.
         */
        final S3Client s3 = S3Client.create();

        /*
         * Set the Amazon S3 bucket name, and then set a lifecycle rule on the
```

```
         * bucket to permanently delete objects 14 days after each object's
         * creation date.
         */
        final LifecycleRule lifeCycleRule = LifecycleRule.builder()
                .expiration(LifecycleExpiration.builder().days(14).build())
                .filter(LifecycleRuleFilter.builder().prefix("").build())
                .status(ExpirationStatus.ENABLED)
                .build();
        final BucketLifecycleConfiguration lifecycleConfig =
BucketLifecycleConfiguration.builder()
                .rules(lifeCycleRule)
                .build();

        // Create the bucket and configure it
        s3.createBucket(CreateBucketRequest.builder().bucket(S3_BUCKET_NAME).build());

s3.putBucketLifecycleConfiguration(PutBucketLifecycleConfigurationRequest.builder()
                .bucket(S3_BUCKET_NAME)
                .lifecycleConfiguration(lifecycleConfig)
                .build());
        System.out.println("Bucket created and configured.");

        // Set the Amazon SQS extended client configuration with large payload support
enabled
        final ExtendedClientConfiguration extendedClientConfig = new
ExtendedClientConfiguration().withPayloadSupportEnabled(s3, S3_BUCKET_NAME);

        final SqsClient sqsExtended = new
AmazonSQSExtendedClient(SqsClient.builder().build(), extendedClientConfig);

        // Create a long string of characters for the message object
        int stringLength = 300000;
        char[] chars = new char[stringLength];
        Arrays.fill(chars, 'x');
        final String myLongString = new String(chars);

        // Create a message queue for this example
        final String queueName = "MyQueue-" + UUID.randomUUID();
        final CreateQueueResponse createQueueResponse =
sqsExtended.createQueue(CreateQueueRequest.builder().queueName(queueName).build());
        final String myQueueUrl = createQueueResponse.queueUrl();
        System.out.println("Queue created.");

        // Send the message
```

```
        final SendMessageRequest sendMessageRequest = SendMessageRequest.builder()
                .queueUrl(myQueueUrl)
                .messageBody(myLongString)
                .build();
        sqsExtended.sendMessage(sendMessageRequest);
        System.out.println("Sent the message.");

        // Receive the message
        final ReceiveMessageResponse receiveMessageResponse =
sqsExtended.receiveMessage(ReceiveMessageRequest.builder().queueUrl(myQueueUrl).build());
        List<Message> messages = receiveMessageResponse.messages();

        // Print information about the message
        for (Message message : messages) {
            System.out.println("\nMessage received.");
            System.out.println("  ID: " + message.messageId());
            System.out.println("  Receipt handle: " + message.receiptHandle());
            System.out.println("  Message body (first 5 characters): " +
message.body().substring(0, 5));
        }

        // Delete the message, the queue, and the bucket
        final String messageReceiptHandle = messages.get(0).receiptHandle();

sqsExtended.deleteMessage(DeleteMessageRequest.builder().queueUrl(myQueueUrl).receiptHandle(me
        System.out.println("Deleted the message.");


sqsExtended.deleteQueue(DeleteQueueRequest.builder().queueUrl(myQueueUrl).build());
        System.out.println("Deleted the queue.");

        deleteBucketAndAllContents(s3);
        System.out.println("Deleted the bucket.");

    }

    private static void deleteBucketAndAllContents(S3Client client) {
        ListObjectsV2Response listObjectsResponse =
client.listObjectsV2(ListObjectsV2Request.builder().bucket(S3_BUCKET_NAME).build());

        listObjectsResponse.contents().forEach(object -> {

client.deleteObject(DeleteObjectRequest.builder().bucket(S3_BUCKET_NAME).key(object.key()).bui
        });
```

```
        ListObjectVersionsResponse listVersionsResponse =
 client.listObjectVersions(ListObjectVersionsRequest.builder().bucket(S3_BUCKET_NAME).build());

        listVersionsResponse.versions().forEach(version -> {

 client.deleteObject(DeleteObjectRequest.builder().bucket(S3_BUCKET_NAME).key(version.key()).ve
        });


 client.deleteBucket(DeleteBucketRequest.builder().bucket(S3_BUCKET_NAME).build());
    }
}
```

Puoi usare Apache Maven per configurare e creare Amazon SQS Extended Client per il tuo progetto Java o per creare l'SDK stesso. Specificate i singoli moduli dall'SDK che utilizzate nella vostra applicazione.

```
<properties>
    <aws-java-sdk.version>2.20.153</aws-java-sdk.version>
</properties>

<dependencies>
    <dependency>
      <groupId>software.amazon.awssdk</groupId>
      <artifactId>sqs</artifactId>
      <version>${aws-java-sdk.version}</version>
    </dependency>
    <dependency>
      <groupId>software.amazon.awssdk</groupId>
      <artifactId>s3</artifactId>
      <version>${aws-java-sdk.version}</version>
    </dependency>
    <dependency>
      <groupId>com.amazonaws</groupId>
      <artifactId>amazon-sqs-java-extended-client-lib</artifactId>
      <version>2.0.4</version>
    </dependency>

    <dependency>
      <groupId>joda-time</groupId>
```

```
      <artifactId>joda-time</artifactId>
      <version>2.12.6</version>
    </dependency>
</dependencies>
```

Gestione di messaggi Amazon SQS di grandi dimensioni con Python e Amazon S3

Puoi utilizzare Amazon Simple Queue Service Extended Client Library for Python e Amazon Simple Storage Service per gestire messaggi Amazon SQS di grandi dimensioni. Ciò è particolarmente utile per consumare carichi utili di messaggi di grandi dimensioni, da 256 KB a 2 GB. La libreria salva il payload del messaggio in un bucket Amazon S3 e invia un messaggio contenente un riferimento dell'oggetto Amazon S3 archiviato a una coda Amazon Amazon SQS.

Puoi usare la Extended Client Library for Python per fare quanto segue:

- Specificare se i payload sono sempre archiviati in Amazon S3 o archiviati solo in S3 quando una dimensione del payload supera i 256 KB
- Invia un messaggio che fa riferimento a un singolo oggetto di messaggio archiviato in un bucket Amazon S3
- Recupera l'oggetto payload corrispondente da un bucket Amazon S3
- Eliminare l'oggetto payload corrispondente da un bucket Amazon S3

Prerequisiti

Di seguito sono riportati i prerequisiti per l'utilizzo della Amazon SQS Extended Client Library for Python:

- Un AWS account con le credenziali necessarie. Per creare un AWS account, vai alla AWShome page, quindi scegli Crea un AWS account. Segui le istruzioni. Per informazioni sulle credenziali, consulta Credenziali.
- Un AWS SDK: l'esempio in questa pagina utilizza AWS Python SDK Boto3. Per installare e configurare l'SDK, consulta la documentazione dell'AWSSDK per Python nella Guida per sviluppatori SDK AWS for Python
- Python 3.x (o successivo) e. `pip`
- La libreria Amazon SQS Extended Client per Python, disponibile da PyPI

> **Note**
>
> Puoi utilizzare Amazon SQS Extended Client Library for Python per gestire i messaggi Amazon SQS utilizzando Amazon S3 solo con l'SDK per Python. AWS Non puoi farlo con la AWS CLI, la console Amazon SQS, l'API HTTP di Amazon SQS o qualsiasi altro SDK. AWS

Configurazione dello storage dei messaggi

Amazon SQS Extended Client utilizza i seguenti attributi dei messaggi per configurare le opzioni di storage dei messaggi di Amazon S3:

- `large_payload_support`: il nome del bucket Amazon S3 per archiviare messaggi di grandi dimensioni.
- `always_through_s3`: Se`True`, allora tutti i messaggi vengono archiviati in Amazon S3. Se`False`, i messaggi di dimensioni inferiori a 256 KB non verranno serializzati nel bucket s3. Il valore predefinito è `False`.
- `use_legacy_attribute`: Se`True`, tutti i messaggi pubblicati utilizzano l'attributo Legacy reserved message (`SQSLargePayloadSize`) anziché l'attuale attributo riservato del messaggio (). `ExtendedPayloadSize`

Gestione di messaggi Amazon SQS di grandi dimensioni con Extended Client Library for Python

L'esempio seguente crea un bucket Amazon S3 con un nome casuale. Quindi crea una coda Amazon SQS denominata `MyQueue` e invia un messaggio che viene archiviato in un bucket S3 e contiene più di 256 KB nella coda. Infine, il codice utilizza il messaggio, restituisce informazioni su di esso e poi elimina il messaggio, la coda e il bucket.

```
import boto3
import sqs_extended_client

#Set the Amazon SQS extended client configuration with large payload.
sqs_extended_client = boto3.client("sqs", region_name="us-east-1")
sqs_extended_client.large_payload_support = "S3_BUCKET_NAME"
sqs_extended_client.use_legacy_attribute = False
```

```
# Create an SQS message queue for this example. Then, extract the queue URL.
queue = sqs_extended_client.create_queue(
    QueueName = "MyQueue"
)
queue_url = sqs_extended_client.get_queue_url(
    QueueName = "MyQueue"
)['QueueUrl']


# Create the S3 bucket and allow message objects to be stored in the bucket.
sqs_extended_client.s3_client.create_bucket(Bucket=sqs_extended_client.large_payload_support)

# Sending a large message
small_message = "s"
large_message = small_message * 300000 # Shall cross the limit of 256 KB

send_message_response = sqs_extended_client.send_message(
    QueueUrl=queue_url,
    MessageBody=large_message
)
assert send_message_response['ResponseMetadata']['HTTPStatusCode'] == 200

# Receiving the large message
receive_message_response = sqs_extended_client.receive_message(
    QueueUrl=queue_url,
    MessageAttributeNames=['All']
)
assert receive_message_response['Messages'][0]['Body'] == large_message
receipt_handle = receive_message_response['Messages'][0]['ReceiptHandle']

# Deleting the large message
# Set to True for deleting the payload from S3
sqs_extended_client.delete_payload_from_s3 = True
delete_message_response = sqs_extended_client.delete_message(
    QueueUrl=queue_url,
    ReceiptHandle=receipt_handle
)

assert delete_message_response['ResponseMetadata']['HTTPStatusCode'] == 200

# Deleting the queue
delete_queue_response = sqs_extended_client.delete_queue(
    QueueUrl=queue_url
```

```
)

assert delete_queue_response['ResponseMetadata']['HTTPStatusCode'] == 200
```

Configurazione delle code Amazon SQS (console)

Usa la console Amazon SQS per configurare e gestire le code e le funzionalità Amazon Simple Queue Service (Amazon SQS). Puoi anche utilizzare la console per configurare funzionalità come la crittografia lato server, associare una coda DLQ alla coda o impostare un trigger per richiamare una funzione AWS Lambda.

Argomenti

Controllo degli accessi basato su attributi (ABAC) con Amazon SQS

Che cos'è ABAC?

Il controllo degli accessi basato su attributi (ABAC) è un processo di autorizzazione che definisce le autorizzazioni in base ai tag che possono essere collegati agli utenti e alle risorse AWS. ABAC fornisce un controllo degli accessi granulare e flessibile basato su attributi e valori, riduce i rischi per la sicurezza legati alle policy riconfigurate basate sui ruoli e centralizza il controllo e la gestione delle policy di accesso. Per ulteriori informazioni su ABAC, consulta Che cos'è ABAC per AWS nella Guida per l'utente di IAM.

Amazon SQS supporta ABAC consentendoti di controllare l'accesso alle code Amazon SQS in base ai tag e agli alias associati a una coda Amazon SQS. Le chiavi di condizione tag e alias che abilitano

ABAC in Amazon SQS autorizzano le entità IAM a utilizzare le code Amazon SQS senza modificare le policy o gestire le concessioni.

Con ABAC, puoi utilizzare i tag per configurare le autorizzazioni e le policy di accesso IAM per le tue code Amazon SQS, il che ti aiuta a dimensionare la gestione delle autorizzazioni. Puoi creare un'unica policy di autorizzazioni in IAM utilizzando i tag che aggiungi a ciascun ruolo aziendale, senza dover aggiornare la policy ogni volta che aggiungi una nuova risorsa. Puoi anche allegare tag ai presidi IAM per creare una policy ABAC. Puoi progettare policy ABAC per consentire le operazioni di Amazon SQS quando il tag sul ruolo utente IAM che effettua la chiamata corrisponde al tag di coda di Amazon SQS. Per ulteriori informazioni sul tagging in AWS, consulta Strategie di tagging AWS e Tag per l'allocazione dei costi Amazon SQS.

> **Note**
>
> ABAC per Amazon SQS è attualmente disponibile in tutte le regioni commerciali AWS in cui è disponibile Amazon SQS, con le seguenti eccezioni:
>
> - Asia Pacific (Hyderabad)
> - Asia Pacifico (Melbourne)
> - Europa (Spagna)
> - Europa (Zurigo)

Qual è il vantaggio di utilizzare ABAC in Amazon SQS?

Ecco alcuni vantaggi dell'utilizzo di ABAC in Amazon SQS:

- ABAC per Amazon SQS richiede un numero minore di policy di autorizzazione. Non è necessario creare policy diverse per diverse mansioni lavorative. Puoi utilizzare tag di risorse e richieste che si applicano a più di una coda, il che riduce il sovraccarico operativo.
- Usa ABAC per dimensionare rapidamente i team. Le autorizzazioni per le nuove risorse vengono concesse automaticamente in base ai tag quando le risorse vengono etichettate in modo appropriato durante la loro creazione.
- Utilizza le autorizzazioni sul principale IAM per limitare l'accesso alle risorse. Puoi creare tag per il principale IAM e utilizzarli per limitare l'accesso ad azioni specifiche che corrispondono ai tag sul principale IAM. Ciò ti aiuta ad automatizzare il processo di concessione delle autorizzazioni per le richieste.

- Tieni traccia di chi accede alle tue risorse. Puoi determinare l'identità di una sessione esaminando gli attributi utente in AWS CloudTrail.

Argomenti

Chiavi di condizione ABAC per Amazon SQS

Puoi utilizzare le seguenti chiavi di condizione per controllare le operazioni delle funzioni:

Chiavi di condizione ABAC	Descrizione	Tipo di policy	Operazioni Amazon SQS
seghe: ResourceTag	Il tag (chiave e valore) sulla chiave Amazon SQS corrisponde al tag (chiave e valore) o al modello di tag nella policy	Solo policy IAM	Operazioni delle risorse di coda Amazon SQS
leggi: RequestTag	Il tag (chiave e valore) nelle operazioni della risorsa della coda corrisponde al tag (chiave e valore) o al modello di tag nella policy	Policy di coda e policy IAM	TagQueue, UntagQueue, CreateQueue
leggi: TagKeys	Le chiavi tag nella richiesta corrispon dono alle chiavi tag nella policy	Policy di coda e policy IAM	TagQueue, UntagQueue, CreateQueue

Assegnazione di tag per il controllo degli accessi

Di seguito è riportato un esempio di come utilizzare i tag per il controllo degli accessi. La policy IAM limita un utente IAM a tutte le azioni Amazon SQS per tutte le code che includono un tag di risorsa con l'ambiente della chiave e la produzione di valore. Per ulteriori informazioni, consulta Controllo dell'accesso basato su attributi con tag e AWS Organizations.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAccessForProd",
      "Effect": "Deny",
      "Action": "sqs:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/environment": "prod"
        }
      }
    }
  ]
}
```

Creazione di utenti IAM e code Amazon SQS

Gli esempi seguenti spiegano come creare una policy ABAC per controllare l'accesso ad Amazon SQS utilizzando la AWS Management Console e AWS CloudFormation.

Utilizzo di AWS Management Console

Crea un utente IAM

1. Accedi a AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Seleziona Utenti dal pannello di navigazione sinistro.
3. Scegli Aggiungi utenti e inserisci un nome nella casella di testo Nome utente.
4. Seleziona Chiave di accesso - Accesso programmatico e scegli Next:Permissions.
5. Scegli Successivo: Tag.

6. Aggiungi un tag con la chiave `environment` e il valore del tag `beta`.
7. Scegli Next:Review, quindi Crea utente.
8. Copiare e archiviare l'ID chiave di accesso e la chiave di accesso segreta in una posizione sicura.

Aggiungi le autorizzazioni degli utenti IAM

1. Seleziona l’utente IAM creato.
2. Scegliere Add inline policy (Aggiungi policy inline).
3. Nella scheda JSON incolla la policy seguente:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessForSameResTag",
      "Effect": "Allow",
      "Action": [
        "sqs:SendMessage",
        "sqs:ReceiveMessage",
        "sqs:DeleteMessage"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/environment": "${aws:PrincipalTag/environment}"
        }
      }
    },
    {
      "Sid": "AllowAccessForSameReqTag",
      "Effect": "Allow",
      "Action": [
        "sqs:CreateQueue",
        "sqs:DeleteQueue",
        "sqs:SetQueueAttributes",
        "sqs:tagqueue"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
```

```
          "aws:RequestTag/environment": "${aws:PrincipalTag/environment}"
        }
      }
    },
    {
      "Sid": "DenyAccessForProd",
      "Effect": "Deny",
      "Action": "sqs:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/stage": "prod"
        }
      }
    }
  ]
}
```

4. Scegli Esamina la policy.
5. Scegli Crea policy.

Uso di AWS CloudFormation

Utilizza il seguente modello AWS CloudFormation di esempio per creare un utente IAM con una policy in linea allegata e una coda Amazon SQS:

```
AWSTemplateFormatVersion: "2010-09-09"
Description: "CloudFormation template to create IAM user with custom inline policy"
Resources:
    IAMPolicy:
        Type: "AWS::IAM::Policy"
        Properties:
            PolicyDocument: |
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Sid": "AllowAccessForSameResTag",
                            "Effect": "Allow",
                            "Action": [
                                "sqs:SendMessage",
                                "sqs:ReceiveMessage",
```

```
                            "sqs:DeleteMessage"
                        ],
                        "Resource": "*",
                        "Condition": {
                            "StringEquals": {
                                "aws:ResourceTag/environment": "${aws:PrincipalTag/
environment}"
                            }
                        }
                    },
                    {
                        "Sid": "AllowAccessForSameReqTag",
                        "Effect": "Allow",
                        "Action": [
                            "sqs:CreateQueue",
                            "sqs:DeleteQueue",
                            "sqs:SetQueueAttributes",
                            "sqs:tagqueue"
                        ],
                        "Resource": "*",
                        "Condition": {
                            "StringEquals": {
                                "aws:RequestTag/environment": "${aws:PrincipalTag/
environment}"
                            }
                        }
                    },
                    {
                        "Sid": "DenyAccessForProd",
                        "Effect": "Deny",
                        "Action": "sqs:*",
                        "Resource": "*",
                        "Condition": {
                            "StringEquals": {
                                "aws:ResourceTag/stage": "prod"
                            }
                        }
                    }
                ]
            }

        Users:
          - "testUser"
        PolicyName: tagQueuePolicy
```

```
    IAMUser:
        Type: "AWS::IAM::User"
        Properties:
            Path: "/"
            UserName: "testUser"
            Tags:
              -
                Key: "environment"
                Value: "beta"
```

Test del controllo dell'accesso basato sugli attributi

Gli esempi seguenti mostrano come testare il controllo degli accessi basato su attributi in Amazon SQS.

Crea una coda con la chiave del tag impostata su environment e il valore del tag impostato su prod

Esegui questo comando AWS CLI per testare la creazione della coda con la chiave del tag impostata su environment e il valore del tag impostato su prod. Se non disponi della AWS CLI, puoi scaricarla e configurarla per la tua macchina.

```
aws sqs create-queue --queue-name prodQueue –region us-east-1 –tags "environment=prod"
```

Ricevi un errore `AccessDenied` dall'endpoint Amazon SQS:

```
An error occurred (AccessDenied) when calling the CreateQueue operation: Access to the
 resource <queueUrl> is denied.
```

Questo perché il valore del tag sull'utente IAM non corrisponde al tag passato nella chiamata `CreateQueue` API. Ricorda che abbiamo applicato un tag all'utente IAM con la chiave impostata su `environment` e il valore impostato su `beta`.

Crea una coda con la chiave del tag impostata su environment e il valore del tag impostato su beta

Esegui questo comando CLI per testare la creazione della coda con la chiave tag impostata su `environment` e il valore del tag impostato su `beta`.

```
aws sqs create-queue --queue-name betaQueue –region us-east-1 –tags "environment=beta"
```

Riceverai un messaggio di conferma dell'avvenuta creazione della coda, simile a quello riportato di seguito.

```
{
"QueueUrl": "<queueUrl>“
}
```

Invio di un messaggio a una coda

Esegui questo comando CLI per testare l'invio di un messaggio a una coda.

```
aws sqs send-message --queue-url <queueUrl> --message-body testMessage
```

La risposta indica che il messaggio è stato recapitato correttamente alla coda di Amazon SQS. L'autorizzazione utente IAM ti consente di inviare un messaggio a una coda con un tag `beta`. La risposta include `MD5OfMessageBody` e `MessageId` contenente il messaggio.

```
{
"MD5OfMessageBody": "<MD5OfMessageBody>",
"MessageId": "<MessageId>"
}
```

Configurazione dei parametri della coda (console)

Quando crei o modifichi una coda, puoi configurare i seguenti parametri:

- Timeout di visibilità: il periodo di tempo in cui un messaggio ricevuto da una coda (da un consumatore) non sarà visibile agli altri utenti di messaggi. Per ulteriori informazioni, consulta Timeout visibilità.

> **Note**
>
> L'utilizzo della console per configurare il timeout di visibilità configura il valore di timeout per tutti i messaggi in coda. Per configurare il timeout per uno o più messaggi, devi utilizzare uno degli SDK AWS.

- Periodo di conservazione dei messaggi: la quantità di tempo in cui Amazon SQS conserva i messaggi che rimangono in coda. Per impostazione predefinita, una coda conserva i messaggi per quattro giorni. È possibile configurare una coda in modo che conservi i messaggi fino a un massimo di 14 giorni. Per altre informazioni, consulta Periodo di conservazione dei messaggi.
- Ritardo di consegna: il periodo di tempo che Amazon SQS ritarderà prima di recapitare un messaggio aggiunto alla coda. Per ulteriori informazioni, consulta Ritardo nella consegna.
- Dimensione massima dei messaggi: la dimensione massima dei messaggi per questa coda. Per ulteriori informazioni consulta Dimensione massima dei messaggi.
- Tempo di attesa per la ricezione dei messaggi: il tempo massimo in cui Amazon SQS attende che i messaggi diventino disponibili dopo che la coda riceve una richiesta di ricezione. Per ulteriori informazioni, consulta Polling brevi e lunghi di Amazon SQS.
- Abilita la deduplicazione basata sui contenuti: Amazon SQS può creare automaticamente ID di deduplicazione in base al corpo del messaggio. Per ulteriori informazioni, consulta Nozioni di base sulle code FIFO di Amazon SQS.
- Abilita FIFO a velocità di trasmissione effettiva elevata: da utilizzare per abilitare una velocità di trasmissione effettiva elevata per i messaggi in coda. La scelta di questa opzione modifica le opzioni correlate (ambito di deduplicazione e limite di velocità di trasmissione effettiva FIFO) con le impostazioni richieste per abilitare una velocità di trasmissione effettiva elevata per le code FIFO. Per ulteriori informazioni, consultare Velocità di trasmissione effettiva elevata per le code FIFO e Quote correlate ai messaggi.
- Policy redrive allow: definisce quali code di origine possono utilizzare questa coda come coda DLQ. Per ulteriori informazioni, consulta Code DLQ di Amazon SQS.

Configurazione dei parametri della coda per una coda esistente (console)

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code). Scegli una coda e seleziona Modifica.
3. Scorri fino alla sezione Configurazione.
4. Per il timeout di visibilità, inserisci la durata e le unità. L'intervallo è compreso tra 0 secondi e 12 ore. Il valore di predefinito è 30 secondi.
5. Per Periodo di conservazione dei messaggi, inserisci la durata e le unità. L'intervallo valido è compreso tra 1 minuto e 14 giorni. Il valore predefinito è 4 giorni.

6. Per una coda standard, inserisci un valore per il tempo di attesa per la ricezione dei messaggi. L'intervallo è compreso tra 0 e 20 secondi. Il valore predefinito è 0 secondi, che imposta uno short polling. Qualsiasi valore diverso da zero imposta un long polling.
7. Per Ritardo di consegna, inserisci la durata e le unità. L'intervallo è compreso tra 30 secondi e 15 minuti. Il valore predefinito è 0 secondi.
8. Per Dimensione massima del messaggio, inserisci un valore. L'intervallo è compreso tra 1 e 256 KB. Il valore predefinito è 256 KB.
9. Per una coda FIFO, scegli Abilita la deduplicazione basata sul contenuto per abilitare la deduplicazione basata sul contenuto. L'impostazione predefinita è disabilitata.
10. (Facoltativo) Affinché una coda FIFO consenta una velocità di trasmissione effettiva più elevata per l'invio e la ricezione di messaggi in coda, seleziona Abilita FIFO ad alta velocità di trasmissione effettiva.

    La scelta di questa opzione modifica le opzioni correlate (ambito di deduplicazione e limite di velocità di trasmissione effettiva FIFO) con le impostazioni richieste per abilitare una velocità di trasmissione effettiva elevata per le code FIFO. Se si modifica una delle impostazioni necessarie per utilizzare FIFO ad alta velocità di trasmissione effettiva, si applica la velocità di trasmissione effettiva normale per la coda e la deduplicazione si verifica come specificato. Per ulteriori informazioni, consultare Velocità di trasmissione effettiva elevata per le code FIFO e Quote correlate ai messaggi.
11. Per la policy Redrive allow, scegli Abilitata. Seleziona una delle seguenti opzioni: Consenti tutto (impostazione predefinita), Per coda o Nega tutto. Quando scegli Per coda, specifica un elenco di un massimo di 10 code di origine in base al nome della risorsa Amazon (ARN).
12. Al termine della configurazione dei parametri della coda, scegli Salva.

Configurazione delle policy d'accesso (Console)

Quando si modifica una coda, è possibile configurarne la policy di accesso.

La policy di accesso definisce gli account, gli utenti e i ruoli che possono accedere alla coda. La policy di accesso definisce anche le operazioni (ad esempio `SendMessage`, `ReceiveMessage` o`DeleteMessage`) a cui gli utenti possono accedere. La policy predefinita consente solo al proprietario della coda di inviare e ricevere messaggi.

Per configurare la policy di accesso per una coda esistente (console)

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code).
3. Scegli una coda e seleziona Modifica.
4. Scorri fino alla sezione Policy di accesso.
5. Modifica le dichiarazioni sulla policy di accesso nella casella di input. Per ulteriori informazioni sulle dichiarazioni relative alle policy di accesso, vedi Identity and Access Management in Amazon SQS.
6. Al termine della configurazione della policy di accesso, scegli Salva.

Configurazione della crittografia lato server (SSE) per una coda tramite chiavi di crittografia gestite da SQS (console)

Oltre all'opzione di crittografia lato server gestita (SSE) predefinita di Amazon SQS, Amazon SQS managed SSE (SSE-SQS) consente di creare una crittografia lato server gestita personalizzata che utilizza chiavi di crittografia gestite da SQS per proteggere i dati sensibili inviati tramite code di messaggi. Con SSE-SQS, non è necessario creare e gestire chiavi di crittografia o modificare il codice per crittografare i dati. SSE-SQS consente di trasmettere i dati in modo sicuro e consente di soddisfare i rigorosi requisiti normativi e di conformità alla crittografia senza costi aggiuntivi.

Per proteggere i dati a riposo, SSE-SQS utilizza la crittografia standard avanzata a 256 bit (AES-256). SSE esegue la crittografia dei messaggi non appena vengono ricevuti da Amazon SQS. Amazon SQS archivia i messaggi in forma crittografata e li decrittografa solo quando li invia a un consumatore autorizzato.

> **Note**
>
> - L'opzione SSE predefinita è efficace solo quando si crea una coda senza specificare gli attributi di crittografia.
> - Amazon SQS consente di disattivare la crittografia di tutte le code. Pertanto, la disattivazione di KMS-SSE non abiliterà automaticamente SQS-SSE. Se desideri abilitare SQS-SSE dopo aver disattivato KMS-SSE, devi aggiungere una modifica dell'attributo nella richiesta.

Per configurare la crittografia SSE-SQS per una coda (console)

> **Note**
>
> Qualsiasi nuova coda creata utilizzando l'endpoint HTTP (non TLS) non abiliterà la crittografia SSE-SQS per impostazione predefinita. È una best practice di sicurezza creare code Amazon SQS utilizzando endpoint HTTPS o Signature Version 4.

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code).
3. Scegli una coda, quindi seleziona Modifica.
4. Espandi crittografia.
5. In Crittografia lato server, scegli Abilita .

   > **Note**
   >
   > Con SSE abilitato, le richieste anonime `SendMessage` e `ReceiveMessage` alla coda crittografata verranno rifiutate. Le best practice di sicurezza di Amazon SQS consigliano di non utilizzare richieste anonime. Se desideri inviare richieste anonime a una coda Amazon SQS, assicurati di disabilitare SSE.

6. Seleziona la chiave Amazon SQS (SSE-SQS). L'utilizzo di questa opzione non prevede costi aggiuntivi.
7. Selezionare Salva.

Configurazione della crittografia lato server (SSE) per una coda (console)

Per proteggere i dati nei messaggi di una coda, Amazon SQS dispone di crittografia lato server (SSE) abilitata di default per tutte le code appena create. Amazon SQS si integra con Amazon Web Services Key Management Service (Amazon Web Services KMS) per gestire le chiavi KMS per la crittografia lato server (SSE). Per ulteriori informazioni sull'uso di SSE, consultare Crittografia a riposo.

La chiave KMS assegnata alla coda deve avere una policy della chiave che includa le autorizzazioni per tutti i principali autorizzati a utilizzare la coda. Per ulteriori informazioni, consulta Gestione delle chiavi.

Se non sei il proprietario della chiave KMS, oppure se effettui l'accesso con un account che non dispone delle autorizzazioni `kms:ListAliases` e `kms:DescribeKey`, non puoi visualizzare le informazioni sulla chiave KMS nella console Amazon SNS. Chiedi al proprietario della chiave KMS di concederti queste autorizzazioni. Per ulteriori informazioni, consulta Gestione delle chiavi.

Quando si crea o si modifica una coda, è possibile configurare SSE-KMS.

Per configurare SSE-KMS per una coda esistente (console)

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code).
3. Scegli una coda, quindi seleziona Modifica.
4. Espandi crittografia.
5. In Crittografia lato server, scegli Abilita .

> **Note**
>
> Con SSE abilitato, le richieste anonime `SendMessage` e `ReceiveMessage` alla coda crittografata verranno rifiutate. Le best practice di sicurezza di Amazon SQS consigliano di non utilizzare richieste anonime. Se desideri inviare richieste anonime a una coda Amazon SQS, assicurati di disabilitare SSE.

6. Seleziona l'opzione Chiave del servizio di gestione delle chiavi AWS (SSE-KMS).

   La console visualizza la descrizione, l'account e l'ARN della chiave KMS della chiave KMS.
7. Specifica l'ID della chiave KMS per la coda. Per ulteriori informazioni, consulta Termini chiave.

   a. Scegli l'opzione Scegli un alias chiave KMS.

   b. La chiave predefinita è la chiave KMS gestita di Amazon Web Services per Amazon SQS. Per utilizzare questa chiave, selezionala dall'elenco delle chiavi KMS.

   c. Per utilizzare una chiave KMS personalizzata dal tuo account Amazon Web Services, selezionala dall'elenco delle chiavi KMS. Per istruzioni sulla creazione di chiavi KMS personalizzate, consulta Creazione delle chiavi nella Guida per sviluppatori di Amazon Web Services Key Management Service.

   d. Per utilizzare una chiave KMS personalizzata non presente nell'elenco o una chiave KMS personalizzata di un altro account Amazon Web Services, scegli Inserisci l'alias della chiave KMS e inserisci il nome della risorsa Amazon (ARN) della chiave KMS.

8. (Facoltativo) Per il periodo di riutilizzo della chiave dati, specifica un valore compreso tra 1 minuto e 24 ore. Il valore predefinito è 5 minuti. Per ulteriori informazioni, consulta Informazioni sul periodo di riutilizzo della chiave di dati.
9. Al termine della configurazione di SSE-KMS, scegli Salva.

Configurazione dei tag di allocazione dei costi per una coda Amazon SQS (console)

Puoi aggiungere i tag per allocazione dei costi alle code Amazon SQS per poterle organizzare e identificare più facilmente. Per ulteriori informazioni, consulta Tag per l'allocazione dei costi Amazon SQS.

Nella pagina Dettagli di una coda, la scheda Tagging mostra i tag per la coda.

Quando si crea o si modifica una coda, è possibile configurare i tag.

Per configurare i tag per una coda esistente (console)

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code).
3. Scegli una coda e seleziona Modifica.
4. Scorri fino alla sezione Tag.
5. Aggiungere, modificare o rimuovere i tag della coda:

   a. Per aggiungere un tag, scegli Aggiungi nuovo tag, inserire una Chiave e un Valore, quindi scegli Applica modifiche.

   b. Per aggiornare un tag, modificane la chiave e il valore.

   c. Per rimuovere un tag, scegli Rimuovi tag accanto a una coppia chiave-valore.

6. Al termine della configurazione dei tag, scegli Salva.

Iscrizione di una coda Amazon SQS a un argomento Amazon SNS (console)

È possibile sottoscrivere una o più code Amazon SQS a un argomento Amazon Simple Notification Service (Amazon SNS). Quando si pubblica un messaggio in un argomento, Amazon SNS invia il messaggio a ogni coda iscritta. Amazon SQS gestisce l'abbonamento e tutte le autorizzazioni necessarie. Per ulteriori informazioni su Amazon SNS, consulta Che cos'è Amazon SNS? nella Guida per gli sviluppatori di Amazon Simple Notification Service.

Quando iscrivi una coda Amazon SQS a un argomento SNS, Amazon SNS usa HTTPS per inoltrare messaggi a Amazon SQS. Per ulteriori informazioni sull'uso di Amazon SNS con code Amazon SQS crittografate, consulta AWS Configura le autorizzazioni KMS per i servizi.

> **Important**
>
> Amazon SQS supporta un massimo di 20 istruzioni per policy di accesso. La sottoscrizione a un argomento Amazon SNS aggiunge una di queste istruzioni. Il superamento di tale importo comporterà la mancata consegna dell'abbonamento all'argomento.

Sottoscrizione di una coda a un argomento SNS (console)

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code).
3. Dall'elenco di code, scegliere la coda per iscriversi all'argomento SNS.
4. Dal menu Operazioni scegliere Subscribe to topic (Sottoscrivi argomento).
5. Da Specifica un argomento Amazon SNS disponibile per questo menu di coda, scegliere Argomento SNS per la coda.

   Se l'argomento SNS non è presente nel menu, scegliere Inserisci ARN dell'argomento Amazon SNS e quindi inserire l'Amazon Resource Name (ARN) dell'argomento.
6. Seleziona Salva.
7. Per verificare il risultato della sottoscrizione, puoi pubblicare nell'argomento e visualizzare il messaggio che l'argomento invia alla coda. Per ulteriori informazioni, consulta Pubblicazione dei messaggi Amazon SNS nella Guida per gli sviluppatori di Amazon Simple Notification Service.

Se la coda di Amazon SQS e l'argomento SNS sono in Account AWS diversi, il proprietario dell'argomento deve prima confermare l'abbonamento. Per ulteriori informazioni, consulta Conferma la sottoscrizione nella Guida per gli sviluppatori di Amazon Simple Notification Service.

Per informazioni sulla sottoscrizione a un argomento SNS interregionale, consulta Invio di messaggi Amazon SNS a una coda Amazon SQS o a una o funzione AWS Lambda in un'altra regione nella Guida per sviluppatori Amazon Simple Notification Service

Configurazione di una coda per attivare una funzione AWS Lambda (console)

Utilizzare una funzione AWS Lambda per elaborare i messaggi da una coda Amazon SQS. Lambda esegue il polling della coda e richiama la funzione Lambda in modo sincrono con un evento che contiene messaggi della coda. Per concedere alla funzione il tempo necessario per elaborare ogni batch dei registri, imposta il timeout visibilità della coda di origine su un tempo pari ad almeno sei volte il timeout configurato per la funzione. Il tempo aggiuntivo consente a Lambda di riprovare se l'esecuzione della funzione viene limitata durante l'elaborazione di un batch precedente.

Puoi specificare un'altra coda che funga da coda DLQ per i messaggi che la tua funzione Lambda non è in grado di elaborare.

Una funzione Lambda può elaborare elementi da più code (un'origine evento Lambda per ogni coda). È possibile utilizzare la stessa coda con più funzioni Lambda.

Se associ una coda crittografata a una funzione Lambda, ma Lambda non esegue il polling dei messaggi, aggiungi l'autorizzazione `kms:Decrypt` al tuo ruolo di esecuzione Lambda.

Sono valide le seguenti limitazioni:

- La coda e la funzione Lambda devono essere nella stessa regione AWS.
- Una coda crittografata che utilizza la chiave predefinita (chiave KMS AWS gestita per Amazon SQS) non può richiamare una funzione Lambda in un'altra Account AWS.

Per informazioni sull'implementazione della funzione Lambda, consulta Utilizzo di AWS Lambda con Amazon SQS nella Guida per sviluppatori di AWS Lambda.

Prerequisiti

Per configurare i trigger delle funzioni Lambda, devi soddisfare i seguenti requisiti:

- Se usi un utente, il ruolo Amazon SQS deve includere le seguenti autorizzazioni:
  - `lambda:CreateEventSourceMapping`
  - `lambda:ListEventSourceMappings`
  - `lambda:ListFunctions`
- Il ruolo di esecuzione Lambda deve includere le seguenti autorizzazioni:
  - `sqs:DeleteMessage`
  - `sqs:GetQueueAttributes`
  - `sqs:ReceiveMessage`
- Se associ una coda crittografata a una funzione Lambda, aggiungi l'autorizzazione `kms:Decrypt` al ruolo di esecuzione Lambda.

Per ulteriori informazioni, consulta Panoramica sulla gestione degli accessi in Amazon SQS.

Per configurare una coda per attivare una funzione Lambda (console)

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code).
3. Nella pagina Code, scegliere la coda da configurare.
4. Nella pagina della coda, scegli la scheda Trigger Lambda.
5. Nella pagina Trigger Lambda, scegli un trigger Lambda.

   Se l'elenco non include il trigger Lambda di cui hai bisogno, scegli Configura il trigger della funzione Lambda. Inserisci il nome della risorsa Amazon (ARN) della funzione Lambda o scegli una risorsa esistente. Quindi scegli Save (Salva).
6. Selezionare Salva. La console salva la configurazione e visualizza la pagina Dettagli per la coda.

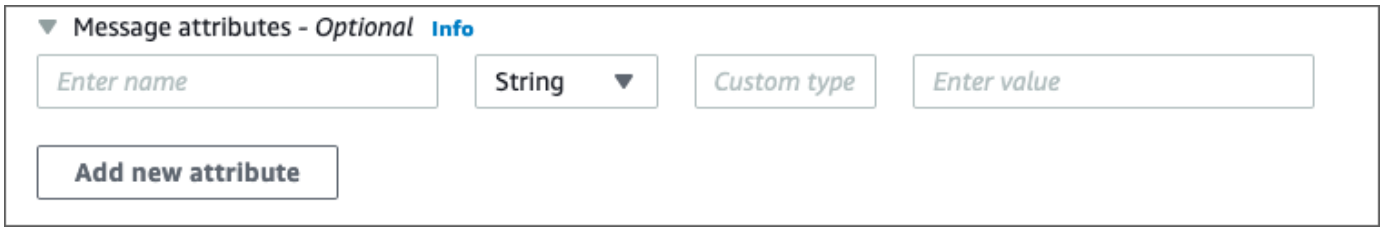
   Nella pagina Dettagli, la scheda Trigger Lambda mostra la funzione Lambda e il relativo stato. Per associare la funzione Lambda alla coda è richiesto circa 1 minuto.
7. Per verificare i risultati della configurazione, è possibile inviare un messaggio alla coda, quindi visualizzare la funzione Lambda attivata nella console Lambda.

Invio di un messaggio con attributi (console)

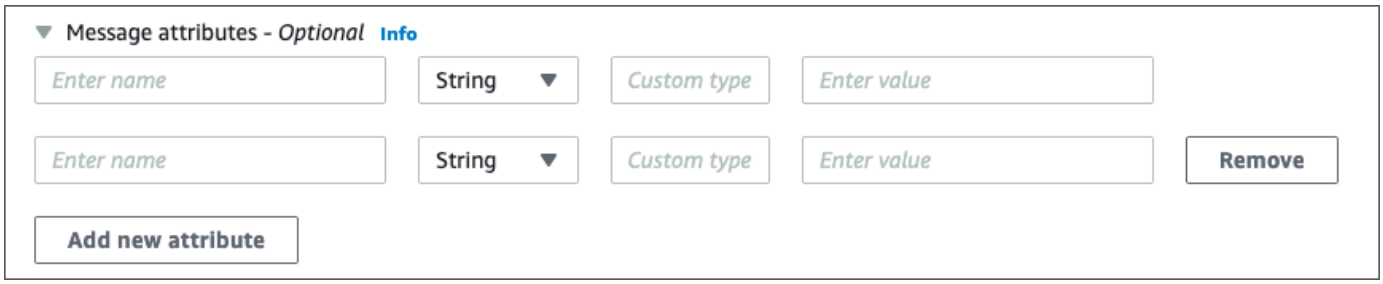
Puoi includere metadati strutturati (come time stamp, dati geospaziali, firme e identificatori) con i messaggi per le code standard e FIFO. Per ulteriori informazioni, consulta Attributi messaggio di Amazon SQS.

Per inviare un messaggio con attributi a una coda (console)

1. Aprire la console Amazon SQS all'indirizzo https://console.aws.amazon.com/sqs/.
2. Nel riquadro di navigazione, scegliere Code (Code).
3. Nella pagina Code, scegliere una coda.
4. Scegli Invia e ricevi messaggi.
5. Inserire i parametri dell'attributo del messaggio.
   a. Nella casella di testo nome, inserire un nome univoco di massimo 256 caratteri.
   b. Per il tipo di attributo, scegliete Stringa, Numero o Binario.
   c. (Facoltativo) Immettere un tipo di dati personalizzato. Ad esempio, puoi aggiungere **`byte`**, **`int`** o **`float`** come tipo di dati personalizzato per Numero.
   d. Nella casella di testo del valore, inserire il valore dell'attributo messaggio.



6. Per aggiungere un altro attributo, scegliere Aggiungi nuovo attributo.



7. È possibile modificare i valori dell'attributo prima di inviare il messaggio.
8. Per eliminare un attributo, scegliere Rimuovi. Per eliminare il primo attributo, chiudere Attributi del messaggio.

9. Una volta terminato di aggiungere gli attributi al messaggio, scegli Invia messaggio. Quando il messaggio viene inviato, la console ne visualizza la conferma. Per visualizzare informazioni sugli attributi del messaggio inviato, scegli Visualizza dettagli. Scegli Fine per chiudere la finestra di dialogo dei dettagli del messaggio.

Best practice per Amazon SQS

Queste best practice possono aiutarti a sfruttare al massimo Amazon SQS.

Argomenti

Consigli per le code FIFO e standard di Amazon SQS

Le seguenti best practice consentono di ridurre i costi ed elaborare i messaggi in modo efficiente utilizzando Amazon SQS.

Argomenti

Lavorare con i messaggi Amazon SQS

Le seguenti indicazioni consentono di elaborare i messaggi in modo efficiente utilizzando Amazon SQS.

Argomenti

Elaborazione di messaggi in modo tempestivo

Le impostazioni del timeout visibilità dipendono da quanto tempo richiede la tua applicazione per elaborare ed eliminare un messaggio. Ad esempio, se la tua applicazione richiede 10 secondi per elaborare un messaggio e imposti il timeout visibilità su 15 minuti, è necessario attendere un tempo relativamente lungo per tentare di elaborare il messaggio di nuovo se il precedente tentativo di elaborazione ha esito negativo. In alternativa, se la tua applicazione richiede 10 secondi per elaborare un messaggio, ma imposti il timeout visibilità su solo 2 secondi, un duplicato del messaggio verrà ricevuto da un altro consumatore mentre il consumatore originale sta ancora lavorando sul messaggio.

Per garantire il tempo sufficiente per elaborare un messaggio, utilizza una delle seguenti strategie:

- Se sai (o puoi ragionevolmente stimare) la quantità di tempo necessaria per elaborare un messaggio, estendi il timeout visibilità del messaggio per il tempo massimo richiesto per elaborare ed eliminare il messaggio. Per ulteriori informazioni, consulta Configurazione del timeout di visibilità.
- Se non sai quanto tempo sia necessario per elaborare un messaggio, crea un heartbeat per il processo del consumatore: specifica il timeout visibilità iniziale (ad esempio, 2 minuti) e poi, se il consumatore lavora ancora sul messaggio, continua a estendere il timeout visibilità di 2 minuti ogni minuto.

> **Important**
>
> Il timeout massimo di visibilità è di 12 ore dal momento in cui Amazon SQS riceve la richiesta `ReceiveMessage`. L'estensione del timeout di visibilità non reimposta il massimo di 12 ore.
> Inoltre, potresti non essere in grado di impostare il timeout per un singolo messaggio per tutte le 12 ore (ad esempio 43.200 secondi) trascorse dalla richiesta `ReceiveMessage` che avvia il timer. Ad esempio, se si riceve un messaggio e si imposta immediatamente il limite massimo di 12 ore inviando una chiamata `ChangeMessageVisibility` della durata di 43.200 secondi, è probabile che la chiamata abbia esito negativo. Tuttavia, l'utilizzo di un valore di 43.195 secondi funzionerà a meno che non vi sia un ritardo significativo tra la richiesta del messaggio tramite `ReceiveMessage` e l'aggiornamento del timeout di visibilità. Se il consumatore ha bisogno di più di 12 ore, prendere in considerazione l'utilizzo di Step Functions.

Gestire gli errori di richiesta

Per gestire gli errori di richiesta, utilizza una delle seguenti strategie:

- Se si utilizza un SDK AWS, avrai già a disposizione un tentativo automatico e la logica di backoff. Per ulteriori informazioni, consulta Ripetizione dei tentativi in caso di errore e backoff esponenziale in AWS nella Riferimenti generali di Amazon Web Services.
- Se non utilizzi le caratteristiche di AWS SDK per nuovi tentativi e backoff, attendi un po' di tempo (ad esempio, 200 ms) prima di ritentare l'operazione ReceiveMessage dopo non aver ricevuto messaggi, un timeout o un messaggio di errore da Amazon SQS. Per il successivo uso di `ReceiveMessage` che offre gli stessi risultati, lascia un po' più di tempo (ad esempio, 400 ms).

Impostazione di polling lungo

Quando il tempo di attesa per l'azione dell'API `ReceiveMessage` è superiore a 0, viene attivato un polling lungo. Il tempo massimo di attesa per il polling lungo è di 20 secondi. Il long polling aiuta a ridurre il costo di utilizzo di Amazon SQS eliminando il numero di risposte vuote (quando non ci sono messaggi disponibili per una richiesta `ReceiveMessage`) e le risposte vuote false (quando i messaggi sono disponibili, ma non sono inclusi nella risposta). Per ulteriori informazioni, consulta Polling brevi e lunghi di Amazon SQS.

Per garantire l'elaborazione ottimale del messaggio, utilizza le seguenti strategie:

- Nella maggior parte dei casi, è possibile impostare il tempo di attesa `ReceiveMessage` su 20 secondi. Se 20 secondi è troppo lungo per la tua applicazione, imposta un tempo di attesa `ReceiveMessage` inferiore (minimo 1 secondo). Se non utilizzi un SDK AWS per accedere ad Amazon SQS, o se configuri un SDK AWS affinché abbia un tempo di attesa più breve, potresti dover modificare il client Amazon SQS per consentire richieste più lunghe o per utilizzare un tempo di attesa più breve per il polling lungo.
- Se implementi il long polling per più code, utilizza un thread per ogni coda invece di un singolo thread per tutte le code. L'utilizzo di un singolo thread per ogni coda consente alla tua applicazione di elaborare i messaggi in ognuna delle code non appena sono disponibili, mentre l'utilizzo di un singolo thread per il polling di più code potrebbe causare la non disponibilità della tua applicazione per elaborare i messaggi disponibili in altre code, mentre l'applicazione attende (fino a 20 secondi) la coda che non ha messaggi disponibili.

> **Important**
>
> Per evitare errori HTTP, assicurarsi che il timeout della risposta HTTP per le richieste `ReceiveMessage` sia più lungo del parametro `WaitTimeSeconds`. Per ulteriori informazioni, consulta ReceiveMessage.

Acquisire messaggi problematici

Per acquisire tutti i messaggi che non possono essere elaborati e per raccogliere parametri parametri CloudWatch accurati, configurare una coda DLQ.

- La policy di reindirizzamento reindirizza i messaggi a una coda DLQ dopo che la coda origine non riesce a elaborare un messaggio per un determinato numero di volte.
- L'utilizzo di una coda DLQ riduce il numero di messaggi e la possibilità di esporti a messaggi poison pill (messaggi che possono essere ricevuti ma non elaborati).
- L'inclusione di un messaggio poison pill in una coda può distorcere il parametro `ApproximateAgeOfOldestMessage` Cloudwatch, fornendo un'età errata del messaggio poison pill. La configurazione di una coda DLQ consente di evitare falsi allarmi durante l'utilizzo di questo parametro.

Impostazione della conservazione di una coda DLQ

Per le code standard, la scadenza di un messaggio si basa sempre sul timestamp della coda originale. Quando un messaggio viene spostato in una coda DLQ, il timestamp della coda rimane invariato. La metrica `ApproximateAgeOfOldestMessage` indica quando il messaggio è stato spostato nella coda DLQ, non quando è stato originariamente inviato. Ad esempio, supponiamo che un messaggio trascorra 1 giorno nella coda originale prima di essere spostato in una coda DLQ. Se il periodo di conservazione della coda DLQ è di 4 giorni, il messaggio viene eliminato dalla coda DLQ dopo 3 giorni e `ApproximateAgeOfOldestMessage`è di tre giorni. Pertanto, è consigliabile impostare sempre il periodo di conservazione di una coda DLQ in modo che sia più lungo del periodo di conservazione della coda originale.

Per le code FIFO, il timestamp della coda si reimposta quando il messaggio viene spostato in una coda DLQ. La metrica `ApproximateAgeOfOldestMessage` indica quando il messaggio è stato spostato nella coda DLQ. Nello stesso esempio precedente, il messaggio viene eliminato dalla coda DLQ dopo 4 giorni e `ApproximateAgeOfOldestMessage` è 4 giorni.

Divieto di elaborazione dei messaggi incoerenti

Poiché Amazon SQS è un sistema distribuito, è possibile per un consumatore non ricevere un messaggio anche quando Amazon SQS contrassegna il messaggio come recapitato mentre restituisce correttamente da una chiamata al metodo API `ReceiveMessage`. In questo caso, Amazon SQS registra il messaggio come consegnato almeno una volta, anche se il consumatore non lo ha mai ricevuto. Poiché non vengono effettuati ulteriori tentativi di recapito dei messaggi in queste condizioni, si sconsiglia di impostare il numero massimo di ricevute su 1 per una dead letter queue.

Implementazione di sistemi Richiesta-Risposta

Quando implementi un sistema richiesta-risposta o di chiamata procedura remota (RPC), tieni presente le seguenti best practice:

- Non creare code di risposta per messaggio. Al contrario, crea le code di risposta all'avvio, per produttore e utilizza un attributo ID messaggio di correlazione per mappare le risposte alle richieste.
- Non lasciare che i produttori condividano le code di risposta. Questo può comportare la ricezione di messaggi di risposta da parte di un produttore destinati a un altro produttore.

Per ulteriori informazioni sull'implementazione del modello di richiesta-risposta utilizzando Temporary Queue Client, consulta Modelli di messaggistica richiesta-risposta (code virtuali).

Riduzione dei costi di Amazon SQS

Le seguenti best practice possono aiutare a ridurre i costi e a sfruttare una potenziale riduzione dei costi aggiuntiva e una risposta quasi istantanea.

Raggruppamento delle azioni con messaggio

Per ridurre i costi, esegui il batch delle azioni con messaggio:

- Per inviare, ricevere ed eliminare messaggi e per modificare il timeout visibilità del messaggio per più messaggi con una singola azione, utilizza le operazioni API in batch di Amazon SQS.
- Per combinare il buffering lato client con il batch delle richieste, utilizza il long polling insieme al client asincrono memorizzato nel buffer incluso con AWS SDK for Java.

> **Note**
>
> L'Amazon SQS Buffered Asynchronous Client attualmente non supporta le code FIFO.

Utilizzo della modalità di polling appropriata

- Il long polling consente di utilizzare messaggi dalla coda Amazon SQS non appena diventano disponibili.
  - Per ridurre il costo di utilizzo di Amazon SQS e ridurre il numero di ricezioni vuote in una coda vuota (risposte all'operazione `ReceiveMessage` che non restituisce messaggi), abilita il long polling. Per ulteriori informazioni, consulta Long Polling Amazon SQS.
  - Per incrementare l'efficienza durante il polling per più thread con più ricezioni, riduci il numero di thread.
  - Il polling lungo è preferibile al polling breve nella maggior parte dei casi.
- Lo short polling restituisce risposte immediatamente, anche se la coda Amazon SQS di cui è stato eseguito il polling è vuota.
  - Per soddisfare i requisiti di un'applicazione che richiede risposte immediate per la richiesta `ReceiveMessage`, utilizza il polling a breve.
  - Il polling breve viene fatturato allo stesso costo del polling lungo.

Trasferimento da una coda standard Amazon SQS a una coda FIFO

Se non hai impostato il parametro `DelaySeconds` per ogni messaggio, puoi passare a una coda FIFO fornendo un ID gruppo di messaggi per ogni messaggio inviato.

Per ulteriori informazioni, consulta Spostamento da una coda standard a una coda FIFO.

Consigli aggiuntivi per le code FIFO di Amazon SQS

Le seguenti best practice consentono di utilizzare l'ID di deduplicazione messaggi e l'ID gruppo di messaggi in modo ottimale. Per ulteriori informazioni, consulta le operazioni `SendMessage` e `SendMessageBatch` nella Documentazione di riferimento delle API di Amazon Simple Queue Service.

Argomenti

Utilizzo dell'ID di deduplicazione dei messaggi Amazon SQS

L'ID di deduplicazione messaggi è il token utilizzato per la deduplicazione dei messaggi inviati. Se un messaggio con un particolare ID di deduplicazione del messaggio viene inviato con successo, tutti i messaggi inviati con lo stesso ID di deduplicazione del messaggio vengono accettati correttamente, ma non vengono recapitati entro l'intervallo di deduplicazione di 5 minuti.

> **Note**
>
> Amazon SQS continua a tenere traccia dell'ID di deduplicazione dei messaggi anche dopo che il messaggio viene ricevuto ed eliminato.

Fornitura dell'ID di deduplicazione messaggi

Il produttore deve fornire i valori di ID di deduplicazione messaggi per ogni messaggio inviato nelle seguenti situazioni:

- I messaggi inviati con corpo del messaggio identico che Amazon SQS deve trattare come univoci.
- I messaggi inviati con contenuti identici ma attributi diversi, che Amazon SQS deve trattare come univoci.
- I messaggi inviati con contenuti diversi (ad esempio, nuovi tentativi inclusi nel corpo del messaggio) che Amazon SQS deve trattare come duplicati.

Abilitazione della deduplicazione per un sistema a consumatore/produttore singolo

Se si dispone di un solo produttore e un singolo consumatore e i messaggi sono specifici perché un ID messaggio specifico dell'applicazione è incluso nel corpo del messaggio, segui queste best practice:

- Abilita la funzionalità di deduplicazione basata sui contenuti per la coda (ciascun messaggio ha un corpo specifico). Il produttore può omettere l'ID di deduplicazione messaggio.

- Anche se il consumatore non è tenuto a fornire un ID del tentativo di richiesta di ricezione per ogni richiesta, è una best practice perché consente un'esecuzione più veloce delle sequenze ritentate dopo esito negativo.
- Puoi riprovare a inviare o ricevere richieste perché non interferiscono con l'ordinazione di messaggi nelle code FIFO.

Progettazione di scenari di recupero interruzioni

Il processo di deduplicazione nelle code FIFO è legata al fattore tempo. Quando si progetta l'applicazione, verificare che sia il produttore sia il consumatore siano in grado di effettuare il recupero in caso di interruzione del client o della rete.

- Il produttore deve essere a conoscenza dell'intervallo di deduplicazione della coda. Amazon SQS dispone di un intervallo di deduplicazione minimo di 5 minuti. La riesecuzione di richieste `SendMessage` dopo la scadenza dell'intervallo di deduplicazione può introdurre messaggi duplicati in coda. Ad esempio, un dispositivo mobile in un auto invia messaggi il cui ordine è importante. Se l'automobile perde connettività cellulare per un periodo di tempo prima di ricevere una conferma, ritentare la richiesta dopo il ritorno della connettività del cellulare può portare alla creazione di un duplicato.
- Il consumatore deve avere un timeout visibilità, che riduce al minimo il rischio di impossibilità di elaborare i messaggi di timeout visibilità prima della scadenza. È possibile estendere la visibilità timeout mentre i messaggi vengono elaborati richiamando l'azione `ChangeMessageVisibility`. Tuttavia, se il timeout visibilità scade, un altro consumatore può immediatamente iniziare a elaborare i messaggi, causando l'elaborazione di un messaggio più volte. Per evitare questo scenario, configura una coda DLQ.

Utilizzo dei timeout di visibilità

Per prestazioni ottimali, imposta il timeout di visibilità su un valore superiore rispetto al timeout di lettura di AWS SDK. Questo si applica all'utilizzo dell'operazione API `ReceiveMessage` con short polling o long polling.

Utilizzo dell'ID gruppo di messaggi Amazon SQS

`MessageGroupId` è il tag che specifica che un messaggio appartiene a un gruppo di messaggi specifico. I messaggi che appartengono allo stesso gruppo di messaggi vengono sempre elaborati

uno per uno, in un ordine rigoroso rispetto al gruppo di messaggi (tuttavia, i messaggi che appartengono a gruppi di messaggi diversi potrebbero essere elaborati in modo errato).

Interleaving di più gruppi di messaggi ordinati

Per eseguire l'interleave di più gruppi di messaggi ordinati all'interno di una singola coda FIFO, utilizza i valori ID del gruppo di messaggi (ad esempio, i dati di sessione per più utenti). In questo scenario, più consumatori sono in grado di elaborare la coda, ma i dati della sessione di ogni utente vengono elaborati in modalità FIFO.

> **Note**
>
> Quando i messaggi appartenenti a un determinato ID gruppo messaggi sono invisibili, gli altri consumatori non sono in grado di elaborare messaggi con lo stesso ID gruppo messaggi.

Evitare l'elaborazione di duplicati in un sistema a più produttori/consumatori

Per evitare l'elaborazione dei messaggi duplicati in un sistema con più produttori e consumatori in cui il throughput e la latenza sono più importanti dell'ordinamento, il produttore deve generare un ID gruppo messaggi univoco per ogni messaggio.

> **Note**
>
> In questo scenario, i duplicati vengono eliminati. Tuttavia, l'ordinazione del messaggio non può essere garantita.
> Qualsiasi scenario con più produttori e consumatori aumenta il rischio di distribuire inavvertitamente un messaggio duplicato nel caso in cui il lavoratore non elabori il messaggio all'interno del timeout visibilità e il messaggio diventi disponibile per un altro lavoratore.

Evitare di avere un backlog di messaggi di grandi dimensioni con lo stesso ID gruppo di messaggi

Per le code FIFO, possono esserci un massimo di 20.000 messaggi in corso (ricevuti da una coda da un consumatore, ma non ancora eliminati dalla coda). Se raggiungi questa quota, Amazon SQS non restituisce alcun messaggio di errore. Una coda FIFO esamina i primi 20.000 messaggi per determinare i gruppi di messaggi disponibili. Ciò significa che se in un singolo gruppo di messaggi

è presente un backlog, non è possibile utilizzare i messaggi provenienti da altri gruppi di messaggi inviati successivamente alla coda fino a quando non si gestisce correttamente il backlog.

> **Note**
>
> Un backlog di messaggi con lo stesso ID gruppo di messaggi potrebbe verificarsi a causa di un consumatore che non è in grado di elaborare correttamente un messaggio. I problemi di elaborazione dei messaggi possono verificarsi a causa di un problema con il contenuto di un messaggio o a causa di un problema tecnico con il consumatore.
> Per spostare i messaggi che non possono essere elaborati ripetutamente e per sbloccare l'elaborazione di altri messaggi con lo stesso ID gruppo di messaggi, valutare la possibilità di configurare una policy di coda DLQ.

Evitare di riutilizzare lo stesso ID gruppo di messaggi con le code virtuali

Per evitare che i messaggi con lo stesso ID gruppo di messaggi inviati a code virtuali diverse con la stessa coda host si blocchino l'un l'altro, evitare di riutilizzare lo stesso ID gruppo di messaggi con le code virtuali.

Utilizzo dell'ID tentativo richiesta di ricezione Amazon SQS

L'ID del tentativo di richiesta di ricezione è il token utilizzato per la deduplicazione delle chiamate `ReceiveMessage`

Durante un'interruzione di rete prolungata che provoca problemi di connettività tra SDK e Amazon SQS, è una best practice fornire l'ID tentativo richiesta di ricezione ed eseguire un nuovo tentativo con lo stesso ID tentativo richiesta di ricezione se l'operazione SDK ha esito negativo.

Esempi di SDK Java di Amazon SQS

È possibile utilizzare AWS SDK for Java per creare applicazioni Java che interagiscono con Amazon Simple Queue Service (Amazon SQS) e altri servizi AWS. Per installare e configurare l'SDK, consulta la Guida introduttiva nella Guida dello sviluppatore di AWS SDK for Java 2.x.

Per esempi di operazioni di coda di Amazon SQS di base, come creare una coda o inviare un messaggio, consulta Lavorare con Amazon SQS Message Queues nella Guida per gli sviluppatori di AWS SDK for Java 2.x.

Gli esempi in questo argomento illustrano funzionalità aggiuntive di Amazon SQS, come la crittografia lato server (SSE), i tag di allocazione dei costi e gli attributi dei messaggi.

Argomenti

- Utilizzo della crittografia lato server (SSE)
- Configurazione di tag per una coda
- Invio degli attributi del messaggio

Utilizzo della crittografia lato server (SSE)

Utilizza AWS SDK for Java per aggiungere la crittografia lato server (SSE) a una coda Amazon SQS. Ogni coda utilizza una chiave AWS Key Management Service (AWS KMS) KMS per generare le chiavi di crittografia dei dati. Questo esempio utilizza la chiave KMS AWS gestita per Amazon SQS. Per ulteriori informazioni su come utilizzare la chiave KMS predefinita, consulta Crittografia a riposo.

Aggiunta di SSE a una coda esistente

Per abilitare la crittografia lato server per una coda esistente, utilizza il metodo `SetQueueAttributes` per impostare l'attributo`KmsMasterKeyId`.

Il seguente esempio di codice imposta AWS KMS key come chiave KMS AWS gestita per Amazon SQS. L'esempio imposta inoltre il periodo di riutilizzo di AWS KMS key su 140 secondi.

Prima di eseguire il codice di esempio, assicurati di aver impostato le credenziali AWS. Per ulteriori informazioni, consulta Configurazione delle credenziali e della regione AWS per lo sviluppo nella Guida per gli sviluppatori di AWS SDK for Java 2.x.

```
// Create an SqsClient for the specified Region.
```

```
SqsClient sqsClient = SqsClient.builder().region(Region.US_WEST_1).build();

// Get the URL of your queue.
String myQueueName = "my queue";
GetQueueUrlResponse getQueueUrlResponse =

 sqsClient.getQueueUrl(GetQueueUrlRequest.builder().queueName(myQueueName).build());
String queueUrl = getQueueUrlResponse.queueUrl();

// Create a hashmap for the attributes. Add the key alias and reuse period to the
 hashmap.
HashMap<QueueAttributeName, String> attributes = new HashMap<QueueAttributeName,
 String>();
final String kmsMasterKeyAlias = "alias/aws/sqs";  // the alias of the AWS managed KMS
 key for Amazon SQS.
attributes.put(QueueAttributeName.KMS_MASTER_KEY_ID, kmsMasterKeyAlias);
attributes.put(QueueAttributeName.KMS_DATA_KEY_REUSE_PERIOD_SECONDS, "140");

// Create the SetQueueAttributesRequest.
SetQueueAttributesRequest set_attrs_request = SetQueueAttributesRequest.builder()
          .queueUrl(queueUrl)
          .attributes(attributes)
          .build();

sqsClient.setQueueAttributes(set_attrs_request);
```

Disabilitazione di SSE per una coda

Per disabilitare la crittografia lato server per una coda esistente, imposta l'attributo `KmsMasterKeyId` su una stringa vuota tramite l'operazione `SetQueueAttributes`.

> **Important**
>
> `null` non è un valore valido per `KmsMasterKeyId`.

Creazione di una coda con SSE

Per abilitare SSE quando crei la coda, aggiungi l'attributo `KmsMasterKeyId` al metodo `CreateQueue` API.

L'esempio seguente spiega come creare una coda con SSE abilitato. La coda utilizza la chiave KMS AWS gestita per Amazon SQS. L'esempio imposta inoltre il periodo di riutilizzo di AWS KMS key su 160 secondi.

Prima di eseguire il codice di esempio, assicurati di aver impostato le credenziali AWS. Per ulteriori informazioni, consulta Configurazione delle credenziali e della regione AWS per lo sviluppo nella Guida per gli sviluppatori di AWS SDK for Java 2.x.

```
// Create an SqsClient for the specified Region.
SqsClient sqsClient = SqsClient.builder().region(Region.US_WEST_1).build();

// Create a hashmap for the attributes. Add the key alias and reuse period to the
 hashmap.
HashMap<QueueAttributeName, String> attributes = new HashMap<QueueAttributeName,
 String>();
final String kmsMasterKeyAlias = "alias/aws/sqs";  // the alias of the AWS managed KMS
 key for Amazon SQS.
attributes.put(QueueAttributeName.KMS_MASTER_KEY_ID, kmsMasterKeyAlias);
attributes.put(QueueAttributeName.KMS_DATA_KEY_REUSE_PERIOD_SECONDS, "140");

// Add the attributes to the CreateQueueRequest.
CreateQueueRequest createQueueRequest =
                CreateQueueRequest.builder()
                        .queueName(queueName)
                        .attributes(attributes)
                        .build();
sqsClient.createQueue(createQueueRequest);
```

Recupero degli attributi SSE

Per informazioni sul recupero degli attributi di coda, consulta Esempi nel riferimento alle API di Amazon Simple Queue Service.

Per recuperare l'ID della chiave KMS o il periodo di riutilizzo della chiave dati per una particolare coda, esegui il metodo `GetQueueAttributes` e recupera i valori `KmsMasterKeyId` e `KmsDataKeyReusePeriodSeconds`.

Configurazione di tag per una coda

Puoi aggiungere i tag di allocazione dei costi alle tue code Amazon SQS per organizzarle e identificarle più facilmente. Gli esempi seguenti mostrano come gestire i tag con AWS SDK for Java. Per ulteriori informazioni, consulta Tag per l'allocazione dei costi Amazon SQS.

Prima di eseguire il codice di esempio, assicurati di aver impostato le credenziali AWS. Per ulteriori informazioni, consulta Configurazione delle credenziali e della regione AWS per lo sviluppo nella Guida per gli sviluppatori di AWS SDK for Java 2.x.

Come elencare i tag

Per elencare i tag di una coda, utilizzate il metodo `ListQueueTags`.

```
// Create an SqsClient for the specified region.
SqsClient sqsClient = SqsClient.builder().region(Region.US_WEST_1).build();

// Get the queue URL.
String queueName = "MyStandardQ1";
GetQueueUrlResponse getQueueUrlResponse =

 sqsClient.getQueueUrl(GetQueueUrlRequest.builder().queueName(queueName).build());
String queueUrl = getQueueUrlResponse.queueUrl();

// Create the ListQueueTagsRequest.
final ListQueueTagsRequest listQueueTagsRequest =

 ListQueueTagsRequest.builder().queueUrl(queueUrl).build();

// Retrieve the list of queue tags and print them.
final ListQueueTagsResponse listQueueTagsResponse =
                                  sqsClient.listQueueTags(listQueueTagsRequest);
System.out.println(String.format("ListQueueTags: \tTags for queue %s are %s.\n",
                queueName, listQueueTagsResponse.tags() ));
```

Aggiunta o aggiornamento dei tag

Per aggiungere o aggiornare i valori dei tag per una coda, utilizzate il metodo `TagQueue`.

```
 // Create an SqsClient for the specified Region.
```

```
SqsClient sqsClient = SqsClient.builder().region(Region.US_WEST_1).build();

// Get the queue URL.
String queueName = "MyStandardQ1";
GetQueueUrlResponse getQueueUrlResponse =

 sqsClient.getQueueUrl(GetQueueUrlRequest.builder().queueName(queueName).build());
String queueUrl = getQueueUrlResponse.queueUrl();

// Build a hashmap of the tags.
final HashMap<String, String> addedTags = new HashMap<>();
        addedTags.put("Team", "Development");
        addedTags.put("Priority", "Beta");
        addedTags.put("Accounting ID", "456def");

//Create the TagQueueRequest and add them to the queue.
final TagQueueRequest tagQueueRequest = TagQueueRequest.builder()
        .queueUrl(queueUrl)
        .tags(addedTags)
        .build();
sqsClient.tagQueue(tagQueueRequest);
```

Rimozione dei tag

Per rimuovere uno o più tag dalla coda, utilizzare il metodo `UntagQueue`. L'esempio seguente rimuove il tag `Accounting ID`.

```

// Create the UntagQueueRequest.
final UntagQueueRequest untagQueueRequest = UntagQueueRequest.builder()
        .queueUrl(queueUrl)
        .tagKeys("Accounting ID")
        .build();

// Remove the tag from this queue.
sqsClient.untagQueue(untagQueueRequest);
```

Invio degli attributi del messaggio

Puoi includere metadati strutturati (come time stamp, dati geospaziali, firme e identificatori) con i messaggi tramite gli attributi dei messaggi. Per ulteriori informazioni, consulta Attributi messaggio di Amazon SQS.

Prima di eseguire il codice di esempio, assicurati di aver impostato le credenziali AWS. Per ulteriori informazioni, consulta Configurazione delle credenziali e della regione AWS per lo sviluppo nella Guida per gli sviluppatori di AWS SDK for Java 2.x.

Definizione degli attributi

Per definire un attributo per un messaggio, aggiungi il codice seguente che utilizza il tipo di dati `MessageAttributeValue`. Per ulteriori informazioni, consulta Componenti attributo del messaggio e Tipi di dati degli attributi di messaggio.

AWS SDK for Java calcola automaticamente i checksum del corpo e degli attributi dei messaggi e li confronta con i dati restituiti da Amazon SQS. Per ulteriori informazioni, consulta la Guida per gli sviluppatori AWS SDK for Java 2.x e Calcolo del digest dei messaggi MD5 per gli attributi di messaggi per altri linguaggi di programmazione.

String

Questo esempio definisce un attributo `String` denominato `Name` con il valore `Jane`.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
messageAttributes.put("Name", new MessageAttributeValue()
.withDataType("String")
.withStringValue("Jane"));
```

Number

Questo esempio definisce un attributo `Number` denominato `AccurateWeight` con il valore `230.00000000000000001`.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
messageAttributes.put("AccurateWeight", new MessageAttributeValue()
.withDataType("Number")
.withStringValue("230.00000000000000001"));
```

Binary

Questo esempio definisce un attributo `Binary` denominato `ByteArray` con il valore di un array non inizializzato di 10 byte.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
messageAttributes.put("ByteArray", new MessageAttributeValue()
.withDataType("Binary")
.withBinaryValue(ByteBuffer.wrap(new byte[10])));
```

String (custom)

Questo esempio definisce l'attributo personalizzato `String.EmployeeId` denominato `EmployeeId` con il valore `ABC123456`.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
messageAttributes.put("EmployeeId", new MessageAttributeValue()
.withDataType("String.EmployeeId")
.withStringValue("ABC123456"));
```

Number (custom)

Questo esempio definisce l'attributo personalizzato `Number.AccountId` denominato `AccountId` con il valore `000123456`.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
messageAttributes.put("AccountId", new MessageAttributeValue()
.withDataType("Number.AccountId")
.withStringValue("000123456"));
```

> **Note**
>
> Poiché il tipo di dati di base è `Number`, il metodo `ReceiveMessage` restituisce `123456`.

Binary (custom)

Questo esempio definisce l'attributo personalizzato `Binary.JPEG` denominato `ApplicationIcon` con il valore di un array non inizializzato di 10 byte.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
```

```
messageAttributes.put("ApplicationIcon", new MessageAttributeValue()
.withDataType("Binary.JPEG")
.withBinaryValue(ByteBuffer.wrap(new byte[10])));
```

Invio di un messaggio con attributi

Questo esempio aggiunge gli attributi a `SendMessageRequest` prima dell'invio del messaggio.

```
// Send a message with an attribute.
final SendMessageRequest sendMessageRequest = new SendMessageRequest();
sendMessageRequest.withMessageBody("This is my message text.");
sendMessageRequest.withQueueUrl(myQueueUrl);
sendMessageRequest.withMessageAttributes(messageAttributes);
sqs.sendMessage(sendMessageRequest);
```

Important

Se invii un messaggio a una coda First-In-First-Out (FIFO), fai in modo che il metodo `sendMessage` venga eseguito dopo aver fornito l'ID gruppo di messaggi.
Se utilizzi l'operazione `SendMessageBatch` invece di `SendMessage`, devi specificare gli attributi per ogni singolo messaggio presente nel batch.

Uso di JMS e Amazon SQS

Amazon SQS Java Messaging Library è un'interfaccia Java Message Service (JMS) per Amazon SQS che consente di sfruttare Amazon SQS in applicazioni che già utilizzano JMS. L'interfaccia consente di usare Amazon SQS come provider JMS, riducendo al minimo le modifiche al codice. Insieme a AWS SDK for Java, la raccolta di messaggistica Amazon SQS Java ti consente di creare connessioni e sessioni JMS, nonché produttori e consumatori in grado di inviare e ricevere messaggi da e verso code Amazon SQS.

La libreria supporta l'invio e la ricezione di messaggi a una coda (modello JMS punto a punto), secondo la specifica JMS 1.1. La libreria supporta l'invio di testo, byte o messaggi oggetto in modo sincrono a code Amazon SQS. La libreria supporta anche la ricezione di oggetti in modo sincrono o asincrono.

Per ulteriori informazioni sulle caratteristiche della raccolta di messaggistica Amazon SQS Java che supportano le specifiche JMS 1.1, consulta Implementazioni JMS 1.1 supportate e le domande frequenti su Amazon SQS.

Argomenti

Prerequisiti

Prima di iniziare, devi disporre dei seguenti requisiti preliminari:

- SDK per Java

  Ci sono due modi per includere l’SDL per Java nel tuo progetto:

  - Scarica e installa l’SDK per Java.
  - Utilizza Maven per ottenere la raccolta di messaggistica Amazon SQS Java.

> **Note**
>
> SDK per Java è incluso come una dipendenza.
> SDK per Java e la libreria client ampia Amazon SQS per Java richiedono J2SE Development Kit 8.0 o versioni successive.

Per informazioni su come scaricare il kit SDK AWS per Java, consulta SDK per Java.

- Raccolta di messaggistica Amazon SQS Java

  Se non utilizzi Maven, devi aggiungere il file del pacchetto `amazon-sqs-java-messaging-lib.jar` al percorso di classe Java. Per informazioni su come scaricare la libreria, consulta raccolta di messaggistica Amazon SQS Java.

  > **Note**
  >
  > La raccolta di messaggistica Amazon SQS Java include il supporto per Maven e Spring Framework.
  > Per esempi di codice che utilizzano Maven, Spring Framework e la raccolta di messaggistica Amazon SQS Java, consulta Esempio Java di utilizzo con JMS e le code standard Amazon SQS.
  >
  > ```
  > <dependency>
  >   <groupId>com.amazonaws</groupId>
  >   <artifactId>amazon-sqs-java-messaging-lib</artifactId>
  >   <version>1.0.4</version>
  >   <type>jar</type>
  > </dependency>
  > ```

- Coda Amazon SQS

  Crea una coda utilizzando AWS Management Console per Amazon SQS, l'API `CreateQueue` o il client Amazon SQS incluso nella raccolta di messaggistica Amazon SQS Java.

  - Per ulteriori informazioni su come creare una coda con Amazon SQS utilizzando AWS Management Console oppure l'API `CreateQueue`, consulta Creazione di una coda.
  - Per informazioni su come utilizzare la raccolta di messaggistica Amazon SQS Java, consulta Nozioni di base sulla raccolta di messaggistica Amazon SQS Java.

Nozioni di base sulla raccolta di messaggistica Amazon SQS Java

Per iniziare a utilizzare il Servizio messaggi Java (JMS) con Amazon SQS, utilizza gli esempi di codice in questa sezione. Le seguenti sezioni illustrano come creare una connessione e una sessione JMS e come inviare e ricevere un messaggio.

L'oggetto client Amazon SQS incluso nella raccolta di messaggistica Amazon SQS Java verifica se esiste una coda Amazon SQS. Se la coda non esiste, il client la crea.

Creazione di una connessione JMS

1. Creare una connection factory e chiamare il metodo `createConnection` rispetto allo stesso.

   ```
   // Create a new connection factory with all defaults (credentials and region) set
    automatically
   SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
           new ProviderConfiguration(),
           AmazonSQSClientBuilder.defaultClient()
           );

   // Create the connection.
   SQSConnection connection = connectionFactory.createConnection();
   ```

   La classe `SQSConnection` estende `javax.jms.Connection`. Insieme ai metodi di connessione JMS standard, `SQSConnection` offre ulteriori metodi, ad esempio `getAmazonSQSClient` e `getWrappedAmazonSQSClient`. Entrambi i metodi consentono di eseguire operazioni di amministrazione non incluse nella specifica JMS, ad esempio la creazione di nuove code. Tuttavia, il metodo `getWrappedAmazonSQSClient` fornisce inoltre una versione integrata del client Amazon SQS utilizzato dalla connessione corrente. Il wrapper trasforma ogni eccezione dal client in una `JMSException`, che può essere utilizzata più facilmente dal codice esistente che si aspetta occorrenze `JMSException`.
2. Puoi utilizzare gli oggetti client restituiti da `getAmazonSQSClient` e `getWrappedAmazonSQSClient` per eseguire operazioni amministrative non incluse nella specifica JMS (ad esempio, è possibile creare una coda Amazon SQS).

   Se disponi di codice esistente che si aspetta eccezioni JMS, devi utilizzare `getWrappedAmazonSQSClient`:

- Se utilizzi `getWrappedAmazonSQSClient`, l'oggetto client restituito trasforma tutte le eccezioni in eccezioni JMS.
- Se utilizzi `getAmazonSQSClient`, le eccezioni sono tutte eccezioni Amazon SQS.

Creazione di una coda Amazon SQS

L'oggetto client integrato verifica se esiste una coda Amazon SQS.

Se la coda non esiste, il client la crea. Se la coda esiste, la funzione non restituisce nulla. Per ulteriori informazioni, consulta la sezione "Creare la coda se necessario" nell'esempio TextMessageSender.java.

Per creare una coda standard

```
// Get the wrapped client
AmazonSQSMessagingClientWrapper client = connection.getWrappedAmazonSQSClient();

// Create an SQS queue named MyQueue, if it doesn't already exist
if (!client.queueExists("MyQueue")) {
    client.createQueue("MyQueue");
}
```

Per creare una coda FIFO

```
// Get the wrapped client
AmazonSQSMessagingClientWrapper client = connection.getWrappedAmazonSQSClient();

// Create an Amazon SQS FIFO queue named MyQueue.fifo, if it doesn't already exist
if (!client.queueExists("MyQueue.fifo")) {
    Map<String, String> attributes = new HashMap<String, String>();
    attributes.put("FifoQueue", "true");
    attributes.put("ContentBasedDeduplication", "true");
    client.createQueue(new
 CreateQueueRequest().withQueueName("MyQueue.fifo").withAttributes(attributes));
}
```

> **Note**
>
> Il nome di una coda FIFO deve terminare con il suffisso `.fifo`.

Per ulteriori informazioni sull'attributo `ContentBasedDeduplication`, consulta Elaborazione "exactly-once".

Invio di messaggi in modo sincrono

1. Quando la connessione e la coda Amazon SQS sottostante sono pronte, creare una sessione JMS nontransacted con la modalità `AUTO_ACKNOWLEDGE`.

```
// Create the nontransacted session with AUTO_ACKNOWLEDGE mode
Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
```

2. Per inviare un messaggio di testo alla coda, creare una coda di identità JMS e un produttore del messaggio.

```
// Create a queue identity and specify the queue name to the session
Queue queue = session.createQueue("MyQueue");

// Create a producer for the 'MyQueue'
MessageProducer producer = session.createProducer(queue);
```

3. Creare un messaggio di testo e inviarlo alla coda.

   - Per inviare un messaggio a una coda standard, non è necessario impostare parametri aggiuntivi.

```
// Create the text message
TextMessage message = session.createTextMessage("Hello World!");

// Send the message
producer.send(message);
System.out.println("JMS Message " + message.getJMSMessageID());
```

   - Per inviare un messaggio a una coda FIFO, è necessario impostare l'ID del gruppo messaggi. Puoi anche impostare un ID di deduplicazione messaggio. Per ulteriori informazioni, consulta Termini chiave.

```
// Create the text message
TextMessage message = session.createTextMessage("Hello World!");

// Set the message group ID
```

```
message.setStringProperty("JMSXGroupID", "Default");

// You can also set a custom message deduplication ID
// message.setStringProperty("JMS_SQS_DeduplicationId", "hello");
// Here, it's not needed because content-based deduplication is enabled for the
 queue

// Send the message
producer.send(message);
System.out.println("JMS Message " + message.getJMSMessageID());
System.out.println("JMS Message Sequence Number " +
 message.getStringProperty("JMS_SQS_SequenceNumber"));
```

Ricezione di messaggi in modo sincrono

1. Per ricevere messaggi, creare un consumatore per la stessa coda e chiamare il metodo `start`.

   Puoi chiamare il metodo `start` sulla connessione in qualsiasi momento. Tuttavia, il consumatore non inizia a ricevere messaggi finché non lo chiami.

   ```
   // Create a consumer for the 'MyQueue'
   MessageConsumer consumer = session.createConsumer(queue);
   // Start receiving incoming messages
   connection.start();
   ```

2. Chiamare il metodo `receive` sul consumatore con un timeout impostati su 1 secondo, quindi stampare i contenuti del messaggio ricevuto.

   - Dopo aver ricevuto un messaggio da una coda standard, puoi accedere al contenuto del messaggio.

     ```
     // Receive a message from 'MyQueue' and wait up to 1 second
     Message receivedMessage = consumer.receive(1000);

     // Cast the received message as TextMessage and display the text
     if (receivedMessage != null) {
         System.out.println("Received: " + ((TextMessage) receivedMessage).getText());
     }
     ```

   - Dopo aver ricevuto un messaggio da una coda FIFO puoi accedere al contenuto del messaggio e ad altri attributi di messaggio specifici FIFO, ad esempio l'ID gruppo di messaggi,

l'ID di deduplicazione messaggio e il numero di sequenza. Per ulteriori informazioni, consulta Termini chiave.

```
// Receive a message from 'MyQueue' and wait up to 1 second
Message receivedMessage = consumer.receive(1000);

// Cast the received message as TextMessage and display the text
if (receivedMessage != null) {
    System.out.println("Received: " + ((TextMessage) receivedMessage).getText());
    System.out.println("Group id: " +
 receivedMessage.getStringProperty("JMSXGroupID"));
    System.out.println("Message deduplication id: " +
 receivedMessage.getStringProperty("JMS_SQS_DeduplicationId"));
    System.out.println("Message sequence number: " +
 receivedMessage.getStringProperty("JMS_SQS_SequenceNumber"));
}
```

3. Chiudere la connessione e la sessione.

```
// Close the connection (and the session).
connection.close();
```

L'esito si presenta in maniera analoga all'immagine riportata di seguito.

```
JMS Message ID:8example-588b-44e5-bbcf-d816example2
Received: Hello World!
```

> **Note**
>
> Puoi utilizzare Spring Framework per inizializzare questi oggetti.
> Per ulteriori informazioni, consulta `SpringExampleConfiguration.xml`, `SpringExample.java` e le altre classi di supporto `ExampleConfiguration.java` e `ExampleCommon.java` nella sezione Esempio Java di utilizzo con JMS e le code standard Amazon SQS.

Per completare esempi di invio e ricezione di oggetti, consulta TextMessageSender.java e SyncMessageReceiver.java.

Ricezione di messaggi in modo asincrono

In questo esempio in Nozioni di base sulla raccolta di messaggistica Amazon SQS Java, viene inviato un messaggio a `MyQueue` e ricevuto in modo sincrono.

L'esempio seguente mostra come ricevere i messaggi in modo asincrono tramite un listener.

1. Implementare l'interfaccia `MessageListener`.

```
class MyListener implements MessageListener {

    @Override
    public void onMessage(Message message) {
        try {
            // Cast the received message as TextMessage and print the text to
 screen.
            System.out.println("Received: " + ((TextMessage) message).getText());
        } catch (JMSException e) {
            e.printStackTrace();
        }
    }
}
```

   Il metodo onMessage dell'interfaccia `MessageListener` viene chiamato quando si riceve un messaggio. In questa implementazione listener, il testo memorizzato nel messaggio è stampato.

2. Invece di chiamare esplicitamente il metodo `receive` sul consumatore, impostare il listener del messaggio del consumatore a un'istanza dell'implementazione `MyListener`. Il thread principale attende per un secondo.

```
// Create a consumer for the 'MyQueue'.
MessageConsumer consumer = session.createConsumer(queue);

// Instantiate and set the message listener for the consumer.
consumer.setMessageListener(new MyListener());

// Start receiving incoming messages.
connection.start();

// Wait for 1 second. The listener onMessage() method is invoked when a message is
 received.
Thread.sleep(1000);
```

Gli altri passaggi sono identici a quelli dell'esempio Nozioni di base sulla raccolta di messaggistica Amazon SQS Java. Per un esempio completo di consumatore asincrono, consulta `AsyncMessageReceiver.java` Esempio Java di utilizzo con JMS e le code standard Amazon SQS.

L'output per questo esempio appare simile al seguente:

```
JMS Message ID:8example-588b-44e5-bbcf-d816example2
Received: Hello World!
```

Utilizzo della modalità di riconoscimento client

L'esempio in Nozioni di base sulla raccolta di messaggistica Amazon SQS Java utilizza la modalità `AUTO_ACKNOWLEDGE` in cui ogni messaggio ricevuto viene confermato automaticamente (e quindi eliminato dalla coda Amazon SQS sottostante).

1. Per riconoscere i messaggi esplicitamente dopo l'elaborazione, è necessario creare la sessione con la modalità `CLIENT_ACKNOWLEDGE`.

   ```
   // Create the non-transacted session with CLIENT_ACKNOWLEDGE mode.
   Session session = connection.createSession(false, Session.CLIENT_ACKNOWLEDGE);
   ```

2. Quando il messaggio viene ricevuto, visualizzarlo e quindi dichiararlo esplicitamente.

   ```
   // Cast the received message as TextMessage and print the text to screen. Also
    acknowledge the message.
   if (receivedMessage != null) {
       System.out.println("Received: " + ((TextMessage) receivedMessage).getText());
       receivedMessage.acknowledge();
       System.out.println("Acknowledged: " + message.getJMSMessageID());
   }
   ```

   > **Note**
   >
   > In questo modo, quando un messaggio viene confermato, tutti i messaggi ricevuti prima di esso sono implicitamente riconosciuti. Ad esempio, se 10 messaggi vengono ricevuti e solo il decimo messaggio viene confermato (nell'ordine in cui i messaggi vengono ricevuti), anche tutti i nove messaggi precedenti vengono riconosciuti.

Gli altri passaggi sono identici a quelli dell'esempio Nozioni di base sulla raccolta di messaggistica Amazon SQS Java. Per un esempio completo di un consumatore sincrono con modalità di riconoscimento client, consulta `SyncMessageReceiverClientAcknowledge.java` Esempio Java di utilizzo con JMS e le code standard Amazon SQS.

L'output per questo esempio appare simile al seguente:

```
JMS Message ID:4example-aa0e-403f-b6df-5e02example5
Received: Hello World!
Acknowledged: ID:4example-aa0e-403f-b6df-5e02example5
```

Utilizzo della modalità di riconoscimento non ordinata

Quando utilizzi la modalità `CLIENT_ACKNOWLEDGE`, tutti i messaggi ricevuti prima di un messaggio esplicitamente riconosciuto sono riconosciuti automaticamente. Per ulteriori informazioni, consulta Utilizzo della modalità di riconoscimento client.

La raccolta di messaggistica Amazon SQS Java fornisce un'altra modalità di conferma. Quando utilizzi la modalità `UNORDERED_ACKNOWLEDGE`, tutti i messaggi ricevuti devono essere individualmente ed esplicitamente riconosciuti dal client, indipendentemente dall'ordine di ricezione. Per eseguire questa operazione, è necessario creare una sessione con la modalità `UNORDERED_ACKNOWLEDGE`.

```
// Create the non-transacted session with UNORDERED_ACKNOWLEDGE mode.
Session session = connection.createSession(false, SQSSession.UNORDERED_ACKNOWLEDGE);
```

Gli altri passaggi sono identici a quelli dell'esempio Utilizzo della modalità di riconoscimento client. Per un esempio completo di consumatore sincrono con la modalità `UNORDERED_ACKNOWLEDGE`, consulta `SyncMessageReceiverUnorderedAcknowledge.java`.

In questo esempio, l'output appare simile al seguente:

```
JMS Message ID:dexample-73ad-4adb-bc6c-4357example7
Received: Hello World!
Acknowledged: ID:dexample-73ad-4adb-bc6c-4357example7
```

Utilizzo del client Servizio messaggi Java (JMS) di Amazon SQS con altri client Amazon SQS

L'utilizzo del client Servizio messaggi Java (JMS) di Amazon SQS con AWS SDK limita la dimensione dei messaggi Amazon SQS a 256 KB. Tuttavia, è possibile creare un provider JMS utilizzando qualsiasi client Amazon SQS. Ad esempio, puoi utilizzare il client JMS con la libreria client ampia di Amazon SQS per Java per inviare un messaggio Amazon SQS che contiene un riferimento a un payload del messaggio (fino a 2 GB) in Amazon S3. Per ulteriori informazioni, consulta Gestione di messaggi Amazon SQS di grandi dimensioni con Java e Amazon S3.

Il seguente codice Java di esempio crea il provider JMS per la libreria client ampia:

```
AmazonS3 s3 = new AmazonS3Client(credentials);
Region s3Region = Region.getRegion(Regions.US_WEST_2);
s3.setRegion(s3Region);

// Set the Amazon S3 bucket name, and set a lifecycle rule on the bucket to
// permanently delete objects a certain number of days after each object's creation
 date.
// Next, create the bucket, and enable message objects to be stored in the bucket.
BucketLifecycleConfiguration.Rule expirationRule = new
 BucketLifecycleConfiguration.Rule();
expirationRule.withExpirationInDays(14).withStatus("Enabled");
BucketLifecycleConfiguration lifecycleConfig = new
 BucketLifecycleConfiguration().withRules(expirationRule);

s3.createBucket(s3BucketName);
s3.setBucketLifecycleConfiguration(s3BucketName, lifecycleConfig);
System.out.println("Bucket created and configured.");

// Set the SQS extended client configuration with large payload support enabled.
ExtendedClientConfiguration extendedClientConfig = new ExtendedClientConfiguration()
    .withLargePayloadSupportEnabled(s3, s3BucketName);

AmazonSQS sqsExtended = new AmazonSQSExtendedClient(new AmazonSQSClient(credentials),
 extendedClientConfig);
Region sqsRegion = Region.getRegion(Regions.US_WEST_2);
sqsExtended.setRegion(sqsRegion);
```

Il seguente codice Java di esempio crea la connection factory:

```
// Create the connection factory using the environment variable credential provider.
// Pass the configured Amazon SQS Extended Client to the JMS connection factory.
SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
        new ProviderConfiguration(),
        sqsExtended
        );

// Create the connection.
SQSConnection connection = connectionFactory.createConnection();
```

Esempio Java di utilizzo con JMS e le code standard Amazon SQS

Negli esempi di codice seguenti viene illustrato come utilizzare Java Message Service (JMS) con le code standard Amazon SQS. Per ulteriori informazioni sull'utilizzo delle code FIFO, consulta Per creare una coda FIFO, Invio di messaggi in modo sincrono e Ricezione di messaggi in modo sincrono. (La ricezione sincrona dei messaggi è la stessa per le code standard e FIFO. Tuttavia, i messaggi nelle code FIFO contengono più attributi).

ExampleConfiguration.java

Il seguente esempio di codice Java SDK v 1.x imposta il nome di coda predefinito, la regione e le credenziali da utilizzare con gli altri esempi Java.

```
/*
 * Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

public class ExampleConfiguration {
    public static final String DEFAULT_QUEUE_NAME = "SQSJMSClientExampleQueue";
```

```
    public static final Region DEFAULT_REGION = Region.getRegion(Regions.US_EAST_2);

    private static String getParameter( String args[], int i ) {
        if( i + 1 >= args.length ) {
            throw new IllegalArgumentException( "Missing parameter for " + args[i] );
        }
        return args[i+1];
    }

    /**
     * Parse the command line and return the resulting config. If the config parsing
fails
     * print the error and the usage message and then call System.exit
     *
     * @param app the app to use when printing the usage string
     * @param args the command line arguments
     * @return the parsed config
     */
    public static ExampleConfiguration parseConfig(String app, String args[]) {
        try {
            return new ExampleConfiguration(args);
        } catch (IllegalArgumentException e) {
            System.err.println( "ERROR: " + e.getMessage() );
            System.err.println();
            System.err.println( "Usage: " + app + " [--queue <queue>] [--region
<region>] [--credentials <credentials>] ");
            System.err.println( "  or" );
            System.err.println( "       " + app + " <spring.xml>" );
            System.exit(-1);
            return null;
        }
    }

    private ExampleConfiguration(String args[]) {
        for( int i = 0; i < args.length; ++i ) {
            String arg = args[i];
            if( arg.equals( "--queue" ) ) {
                setQueueName(getParameter(args, i));
                i++;
            } else if( arg.equals( "--region" ) ) {
                String regionName = getParameter(args, i);
                try {
                    setRegion(Region.getRegion(Regions.fromName(regionName)));
```

```
                } catch( IllegalArgumentException e ) {
                    throw new IllegalArgumentException( "Unrecognized region " +
regionName );
                }
                i++;
            } else if( arg.equals( "--credentials" ) ) {
                String credsFile = getParameter(args, i);
                try {
                    setCredentialsProvider( new
PropertiesFileCredentialsProvider(credsFile) );
                } catch (AmazonClientException e) {
                    throw new IllegalArgumentException("Error reading credentials from
" + credsFile, e );
                }
                i++;
            } else {
                throw new IllegalArgumentException("Unrecognized option " + arg);
            }
        }
    }

    private String queueName = DEFAULT_QUEUE_NAME;
    private Region region = DEFAULT_REGION;
    private AWSCredentialsProvider credentialsProvider = new
DefaultAWSCredentialsProviderChain();

    public String getQueueName() {
        return queueName;
    }

    public void setQueueName(String queueName) {
        this.queueName = queueName;
    }

    public Region getRegion() {
        return region;
    }

    public void setRegion(Region region) {
        this.region = region;
    }

    public AWSCredentialsProvider getCredentialsProvider() {
        return credentialsProvider;
```

```
    }

    public void setCredentialsProvider(AWSCredentialsProvider credentialsProvider) {
        // Make sure they're usable first
        credentialsProvider.getCredentials();
        this.credentialsProvider = credentialsProvider;
    }
}
```

TextMessageSender.java

Il seguente codice Java crea un produttore del messaggio di testo.

```
/*
 * Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

public class TextMessageSender {
    public static void main(String args[]) throws JMSException {
        ExampleConfiguration config =
 ExampleConfiguration.parseConfig("TextMessageSender", args);

        ExampleCommon.setupLogging();

        // Create the connection factory based on the config
        SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
                new ProviderConfiguration(),
                AmazonSQSClientBuilder.standard()
                        .withRegion(config.getRegion().getName())
                        .withCredentials(config.getCredentialsProvider())
                );
```

```
        // Create the connection
        SQSConnection connection = connectionFactory.createConnection();

        // Create the queue if needed
        ExampleCommon.ensureQueueExists(connection, config.getQueueName());

        // Create the session
        Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
        MessageProducer producer =
session.createProducer( session.createQueue( config.getQueueName() ) );

        sendMessages(session, producer);

        // Close the connection. This closes the session automatically
        connection.close();
        System.out.println( "Connection closed" );
    }

    private static void sendMessages( Session session, MessageProducer producer ) {
        BufferedReader inputReader = new BufferedReader(
            new InputStreamReader( System.in, Charset.defaultCharset() ) );

        try {
            String input;
            while( true ) {
                System.out.print( "Enter message to send (leave empty to exit): " );
                input = inputReader.readLine();
                if( input == null || input.equals("" ) ) break;

                TextMessage message = session.createTextMessage(input);
                producer.send(message);
                System.out.println( "Send message " + message.getJMSMessageID() );
            }
        } catch (EOFException e) {
            // Just return on EOF
        } catch (IOException e) {
            System.err.println( "Failed reading input: " + e.getMessage() );
        } catch (JMSException e) {
            System.err.println( "Failed sending message: " + e.getMessage() );
            e.printStackTrace();
        }
    }
```

```
}
```

SyncMessageReceiver.java

Il seguente codice Java crea un consumatore del messaggio di testo sincrono.

```
/*
 * Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

public class SyncMessageReceiver {
public static void main(String args[]) throws JMSException {
    ExampleConfiguration config =
 ExampleConfiguration.parseConfig("SyncMessageReceiver", args);

    ExampleCommon.setupLogging();

    // Create the connection factory based on the config
    SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
            new ProviderConfiguration(),
            AmazonSQSClientBuilder.standard()
                    .withRegion(config.getRegion().getName())
                    .withCredentials(config.getCredentialsProvider())
            );

    // Create the connection
    SQSConnection connection = connectionFactory.createConnection();

    // Create the queue if needed
    ExampleCommon.ensureQueueExists(connection, config.getQueueName());
```

```
    // Create the session
    Session session = connection.createSession(false, Session.CLIENT_ACKNOWLEDGE);
    MessageConsumer consumer =
 session.createConsumer( session.createQueue( config.getQueueName() ) );

    connection.start();

    receiveMessages(session, consumer);

    // Close the connection. This closes the session automatically
    connection.close();
    System.out.println( "Connection closed" );
}

private static void receiveMessages( Session session, MessageConsumer consumer ) {
    try {
        while( true ) {
            System.out.println( "Waiting for messages");
            // Wait 1 minute for a message
            Message message = consumer.receive(TimeUnit.MINUTES.toMillis(1));
            if( message == null ) {
                System.out.println( "Shutting down after 1 minute of silence" );
                break;
            }
            ExampleCommon.handleMessage(message);
            message.acknowledge();
            System.out.println( "Acknowledged message " + message.getJMSMessageID() );
        }
    } catch (JMSException e) {
        System.err.println( "Error receiving from SQS: " + e.getMessage() );
        e.printStackTrace();
    }
}
}
```

AsyncMessageReceiver.java

Il seguente codice Java crea un consumatore del messaggio di testo asincrono.

```
/*
 * Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
```

```
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

public class AsyncMessageReceiver {
    public static void main(String args[]) throws JMSException, InterruptedException {
        ExampleConfiguration config =
 ExampleConfiguration.parseConfig("AsyncMessageReceiver", args);

        ExampleCommon.setupLogging();

        // Create the connection factory based on the config
        SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
                new ProviderConfiguration(),
                AmazonSQSClientBuilder.standard()
                        .withRegion(config.getRegion().getName())
                        .withCredentials(config.getCredentialsProvider())
                );

        // Create the connection
        SQSConnection connection = connectionFactory.createConnection();

        // Create the queue if needed
        ExampleCommon.ensureQueueExists(connection, config.getQueueName());

        // Create the session
        Session session = connection.createSession(false, Session.CLIENT_ACKNOWLEDGE);
        MessageConsumer consumer =
 session.createConsumer( session.createQueue( config.getQueueName() ) );

        // No messages are processed until this is called
        connection.start();

        ReceiverCallback callback = new ReceiverCallback();
        consumer.setMessageListener( callback );
```

```
        callback.waitForOneMinuteOfSilence();
        System.out.println( "Returning after one minute of silence" );

        // Close the connection. This closes the session automatically
        connection.close();
        System.out.println( "Connection closed" );
    }


    private static class ReceiverCallback implements MessageListener {
        // Used to listen for message silence
        private volatile long timeOfLastMessage = System.nanoTime();

        public void waitForOneMinuteOfSilence() throws InterruptedException {
            for(;;) {
                long timeSinceLastMessage = System.nanoTime() - timeOfLastMessage;
                long remainingTillOneMinuteOfSilence =
                    TimeUnit.MINUTES.toNanos(1) - timeSinceLastMessage;
                if( remainingTillOneMinuteOfSilence < 0 ) {
                    break;
                }
                TimeUnit.NANOSECONDS.sleep(remainingTillOneMinuteOfSilence);
            }
        }


        @Override
        public void onMessage(Message message) {
            try {
                ExampleCommon.handleMessage(message);
                message.acknowledge();
                System.out.println( "Acknowledged message " +
 message.getJMSMessageID() );
                timeOfLastMessage = System.nanoTime();
            } catch (JMSException e) {
                System.err.println( "Error processing message: " + e.getMessage() );
                e.printStackTrace();
            }
        }
    }
}
```

SyncMessageReceiverClientAcknowledge.java

Il seguente esempio di codice Java crea un consumatore asincrono con modalità di riconoscimento client.

```
/*
 * Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

/**
 * An example class to demonstrate the behavior of CLIENT_ACKNOWLEDGE mode for received
 messages. This example
 * complements the example given in {@link SyncMessageReceiverUnorderedAcknowledge} for
 UNORDERED_ACKNOWLEDGE mode.
 *
 * First, a session, a message producer, and a message consumer are created. Then, two
 messages are sent. Next, two messages
 * are received but only the second one is acknowledged. After waiting for the
 visibility time out period, an attempt to
 * receive another message is made. It's shown that no message is returned for this
 attempt since in CLIENT_ACKNOWLEDGE mode,
 * as expected, all the messages prior to the acknowledged messages are also
 acknowledged.
 *
 * This ISN'T the behavior for UNORDERED_ACKNOWLEDGE mode. Please see {@link
 SyncMessageReceiverUnorderedAcknowledge}
 * for an example.
 */
public class SyncMessageReceiverClientAcknowledge {
```

```
   // Visibility time-out for the queue. It must match to the one set for the queue
for this example to work.
   private static final long TIME_OUT_SECONDS = 1;

   public static void main(String args[]) throws JMSException, InterruptedException {
       // Create the configuration for the example
       ExampleConfiguration config =
ExampleConfiguration.parseConfig("SyncMessageReceiverClientAcknowledge", args);

       // Setup logging for the example
       ExampleCommon.setupLogging();

       // Create the connection factory based on the config
       SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
               new ProviderConfiguration(),
               AmazonSQSClientBuilder.standard()
                       .withRegion(config.getRegion().getName())
                       .withCredentials(config.getCredentialsProvider())
               );

       // Create the connection
       SQSConnection connection = connectionFactory.createConnection();

       // Create the queue if needed
       ExampleCommon.ensureQueueExists(connection, config.getQueueName());

       // Create the session  with client acknowledge mode
       Session session = connection.createSession(false, Session.CLIENT_ACKNOWLEDGE);

       // Create the producer and consume
       MessageProducer producer =
session.createProducer(session.createQueue(config.getQueueName()));
       MessageConsumer consumer =
session.createConsumer(session.createQueue(config.getQueueName()));

       // Open the connection
       connection.start();

       // Send two text messages
       sendMessage(producer, session, "Message 1");
       sendMessage(producer, session, "Message 2");

       // Receive a message and don't acknowledge it
       receiveMessage(consumer, false);
```

```
        // Receive another message and acknowledge it
        receiveMessage(consumer, true);

        // Wait for the visibility time out, so that unacknowledged messages reappear
in the queue
        System.out.println("Waiting for visibility timeout...");
        Thread.sleep(TimeUnit.SECONDS.toMillis(TIME_OUT_SECONDS));

        // Attempt to receive another message and acknowledge it. This results in
receiving no messages since
        // we have acknowledged the second message. Although we didn't explicitly
acknowledge the first message,
        // in the CLIENT_ACKNOWLEDGE mode, all the messages received prior to the
explicitly acknowledged message
        // are also acknowledged. Therefore, we have implicitly acknowledged the first
message.
        receiveMessage(consumer, true);

        // Close the connection. This closes the session automatically
        connection.close();
        System.out.println("Connection closed.");
    }

    /**
     * Sends a message through the producer.
     *
     * @param producer Message producer
     * @param session Session
     * @param messageText Text for the message to be sent
     * @throws JMSException
     */
    private static void sendMessage(MessageProducer producer, Session session, String
messageText) throws JMSException {
        // Create a text message and send it
        producer.send(session.createTextMessage(messageText));
    }

    /**
     * Receives a message through the consumer synchronously with the default timeout
(TIME_OUT_SECONDS).
     * If a message is received, the message is printed. If no message is received,
"Queue is empty!" is
     * printed.
```

```
     *
     * @param consumer Message consumer
     * @param acknowledge If true and a message is received, the received message is
 acknowledged.
     * @throws JMSException
     */
    private static void receiveMessage(MessageConsumer consumer, boolean acknowledge)
 throws JMSException {
        // Receive a message
        Message message =
 consumer.receive(TimeUnit.SECONDS.toMillis(TIME_OUT_SECONDS));

        if (message == null) {
            System.out.println("Queue is empty!");
        } else {
            // Since this queue has only text messages, cast the message object and
 print the text
            System.out.println("Received: " + ((TextMessage) message).getText());

            // Acknowledge the message if asked
            if (acknowledge) message.acknowledge();
        }
    }
}
```

SyncMessageReceiverUnorderedAcknowledge.java

Il seguente esempio di codice Java crea un consumatore asincrono con modalità di riconoscimento non ordinato.

```
/*
 * Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
```

```
 *
 */

/**
 * An example class to demonstrate the behavior of UNORDERED_ACKNOWLEDGE mode for
 received messages. This example
 * complements the example given in {@link SyncMessageReceiverClientAcknowledge} for
 CLIENT_ACKNOWLEDGE mode.
 *
 * First, a session, a message producer, and a message consumer are created. Then, two
 messages are sent. Next, two messages
 * are received but only the second one is acknowledged. After waiting for the
 visibility time out period, an attempt to
 * receive another message is made. It's shown that the first message received in the
 prior attempt is returned again
 * for the second attempt. In UNORDERED_ACKNOWLEDGE mode, all the messages must be
 explicitly acknowledged no matter what
 * the order they're received.
 *
 * This ISN'T the behavior for CLIENT_ACKNOWLEDGE mode. Please see {@link
 SyncMessageReceiverClientAcknowledge}
 * for an example.
 */
public class SyncMessageReceiverUnorderedAcknowledge {

    // Visibility time-out for the queue. It must match to the one set for the queue
 for this example to work.
    private static final long TIME_OUT_SECONDS = 1;

    public static void main(String args[]) throws JMSException, InterruptedException {
        // Create the configuration for the example
        ExampleConfiguration config =
 ExampleConfiguration.parseConfig("SyncMessageReceiverUnorderedAcknowledge", args);

        // Setup logging for the example
        ExampleCommon.setupLogging();

        // Create the connection factory based on the config
        SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
                new ProviderConfiguration(),
                AmazonSQSClientBuilder.standard()
                        .withRegion(config.getRegion().getName())
                        .withCredentials(config.getCredentialsProvider())
                );
```

```
        // Create the connection
        SQSConnection connection = connectionFactory.createConnection();

        // Create the queue if needed
        ExampleCommon.ensureQueueExists(connection, config.getQueueName());

        // Create the session  with unordered acknowledge mode
        Session session = connection.createSession(false,
SQSSession.UNORDERED_ACKNOWLEDGE);

        // Create the producer and consume
        MessageProducer producer =
session.createProducer(session.createQueue(config.getQueueName()));
        MessageConsumer consumer =
session.createConsumer(session.createQueue(config.getQueueName()));

        // Open the connection
        connection.start();

        // Send two text messages
        sendMessage(producer, session, "Message 1");
        sendMessage(producer, session, "Message 2");

        // Receive a message and don't acknowledge it
        receiveMessage(consumer, false);

        // Receive another message and acknowledge it
        receiveMessage(consumer, true);

        // Wait for the visibility time out, so that unacknowledged messages reappear
in the queue
        System.out.println("Waiting for visibility timeout...");
        Thread.sleep(TimeUnit.SECONDS.toMillis(TIME_OUT_SECONDS));

        // Attempt to receive another message and acknowledge it. This results in
receiving the first message since
        // we have acknowledged only the second message. In the UNORDERED_ACKNOWLEDGE
mode, all the messages must
        // be explicitly acknowledged.
        receiveMessage(consumer, true);

        // Close the connection. This closes the session automatically
        connection.close();
```

```
        System.out.println("Connection closed.");
    }

    /**
     * Sends a message through the producer.
     *
     * @param producer Message producer
     * @param session Session
     * @param messageText Text for the message to be sent
     * @throws JMSException
     */
    private static void sendMessage(MessageProducer producer, Session session, String
messageText) throws JMSException {
        // Create a text message and send it
        producer.send(session.createTextMessage(messageText));
    }

    /**
     * Receives a message through the consumer synchronously with the default timeout
(TIME_OUT_SECONDS).
     * If a message is received, the message is printed. If no message is received,
"Queue is empty!" is
     * printed.
     *
     * @param consumer Message consumer
     * @param acknowledge If true and a message is received, the received message is
acknowledged.
     * @throws JMSException
     */
    private static void receiveMessage(MessageConsumer consumer, boolean acknowledge)
throws JMSException {
        // Receive a message
        Message message =
consumer.receive(TimeUnit.SECONDS.toMillis(TIME_OUT_SECONDS));

        if (message == null) {
            System.out.println("Queue is empty!");
        } else {
            // Since this queue has only text messages, cast the message object and
print the text
            System.out.println("Received: " + ((TextMessage) message).getText());

            // Acknowledge the message if asked
            if (acknowledge) message.acknowledge();
```

```
        }
    }
}
```

SpringExampleConfiguration.xml

L'esempio di codice XML riportato di seguito è un file di configurazione bean per SpringExample.java.

```
<!--
    Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.

    Licensed under the Apache License, Version 2.0 (the "License").
    You may not use this file except in compliance with the License.
    A copy of the License is located at

    https://aws.amazon.com/apache2.0

    or in the "license" file accompanying this file. This file is distributed
    on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
    express or implied. See the License for the specific language governing
    permissions and limitations under the License.
-->

<?xml version="1.0" encoding="UTF-8"?>
<beans
    xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:util="http://www.springframework.org/schema/util"
    xmlns:p="http://www.springframework.org/schema/p"
    xsi:schemaLocation="
        http://www.springframework.org/schema/beans http://www.springframework.org/
schema/beans/spring-beans-3.0.xsd
        http://www.springframework.org/schema/util http://www.springframework.org/
schema/util/spring-util-3.0.xsd
    ">

    <bean id="CredentialsProviderBean"
 class="com.amazonaws.auth.DefaultAWSCredentialsProviderChain"/>

    <bean id="ClientBuilder" class="com.amazonaws.services.sqs.AmazonSQSClientBuilder"
 factory-method="standard">
        <property name="region" value="us-east-2"/>
        <property name="credentials" ref="CredentialsProviderBean"/>
```

```
    </bean>

    <bean id="ProviderConfiguration"
 class="com.amazon.sqs.javamessaging.ProviderConfiguration">
        <property name="numberOfMessagesToPrefetch" value="5"/>
    </bean>

    <bean id="ConnectionFactory"
 class="com.amazon.sqs.javamessaging.SQSConnectionFactory">
        <constructor-arg ref="ProviderConfiguration" />
        <constructor-arg ref="ClientBuilder" />
    </bean>

    <bean id="Connection" class="javax.jms.Connection"
        factory-bean="ConnectionFactory"
        factory-method="createConnection"
        init-method="start"
        destroy-method="close" />

    <bean id="QueueName" class="java.lang.String">
        <constructor-arg value="SQSJMSClientExampleQueue"/>
    </bean>
</beans>
```

SpringExample.java

Il seguente codice di esempio Java utilizza il file di configurazione bean per inizializzare i tuoi oggetti.

```
/*
 * Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */
```

```
public class SpringExample {
    public static void main(String args[]) throws JMSException {
        if( args.length != 1 || !args[0].endsWith(".xml")) {
            System.err.println( "Usage: " + SpringExample.class.getName() + " <spring
 config.xml>" );
            System.exit(1);
        }

        File springFile = new File( args[0] );
        if( !springFile.exists() || !springFile.canRead() ) {
            System.err.println( "File " + args[0] + " doesn't exist or isn't
 readable.");
            System.exit(2);
        }

        ExampleCommon.setupLogging();

        FileSystemXmlApplicationContext context =
            new FileSystemXmlApplicationContext( "file://" +
 springFile.getAbsolutePath() );

        Connection connection;
        try {
            connection = context.getBean(Connection.class);
        } catch( NoSuchBeanDefinitionException e ) {
            System.err.println( "Can't find the JMS connection to use: " +
 e.getMessage() );
            System.exit(3);
            return;
        }

        String queueName;
        try {
            queueName = context.getBean("QueueName", String.class);
        } catch( NoSuchBeanDefinitionException e ) {
            System.err.println( "Can't find the name of the queue to use: " +
 e.getMessage() );
            System.exit(3);
            return;
        }

        if( connection instanceof SQSConnection ) {
            ExampleCommon.ensureQueueExists( (SQSConnection) connection, queueName );
```

```
        }

        // Create the session
        Session session = connection.createSession(false, Session.CLIENT_ACKNOWLEDGE);
        MessageConsumer consumer =
 session.createConsumer( session.createQueue( queueName) );

        receiveMessages(session, consumer);

        // The context can be setup to close the connection for us
        context.close();
        System.out.println( "Context closed" );
    }

    private static void receiveMessages( Session session, MessageConsumer consumer ) {
        try {
            while( true ) {
                System.out.println( "Waiting for messages");
                // Wait 1 minute for a message
                Message message = consumer.receive(TimeUnit.MINUTES.toMillis(1));
                if( message == null ) {
                    System.out.println( "Shutting down after 1 minute of silence" );
                    break;
                }
                ExampleCommon.handleMessage(message);
                message.acknowledge();
                System.out.println( "Acknowledged message" );
            }
        } catch (JMSException e) {
            System.err.println( "Error receiving from SQS: " + e.getMessage() );
            e.printStackTrace();
        }
    }
}
```

ExampleCommon.java

Il seguente codice di esempio Java verifica se una coda Amazon SQS esiste e quindi ne crea una se non esiste. Include anche il codice di registrazione di esempio.

```
/*
 * Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
```

```
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

public class ExampleCommon {
    /**
     * A utility function to check the queue exists and create it if needed. For most
     * use cases this is usually done by an administrator before the application is
 run.
     */
    public static void ensureQueueExists(SQSConnection connection, String queueName)
 throws JMSException {
        AmazonSQSMessagingClientWrapper client =
 connection.getWrappedAmazonSQSClient();

        /**
         * In most cases, you can do this with just a createQueue call, but
 GetQueueUrl
         * (called by queueExists) is a faster operation for the common case where the
 queue
         * already exists. Also many users and roles have permission to call
 GetQueueUrl
         * but don't have permission to call CreateQueue.
         */
        if( !client.queueExists(queueName) ) {
            client.createQueue( queueName );
        }
    }

    public static void setupLogging() {
        // Setup logging
        BasicConfigurator.configure();
        Logger.getRootLogger().setLevel(Level.WARN);
    }
```

```
    public static void handleMessage(Message message) throws JMSException {
        System.out.println( "Got message " + message.getJMSMessageID() );
        System.out.println( "Content: ");
        if( message instanceof TextMessage ) {
            TextMessage txtMessage = ( TextMessage ) message;
            System.out.println( "\t" + txtMessage.getText() );
        } else if( message instanceof BytesMessage ){
            BytesMessage byteMessage = ( BytesMessage ) message;
            // Assume the length fits in an int - SQS only supports sizes up to 256k so
 that
            // should be true
            byte[] bytes = new byte[(int)byteMessage.getBodyLength()];
            byteMessage.readBytes(bytes);
            System.out.println( "\t" +  Base64.encodeAsString( bytes ) );
        } else if( message instanceof ObjectMessage ) {
            ObjectMessage objMessage = (ObjectMessage) message;
            System.out.println( "\t" + objMessage.getObject() );
        }
    }
}
```

Implementazioni JMS 1.1 supportate

La raccolta di messaggistica Amazon SQS Java supporta le seguenti implementazioni JMS 1.1. Per ulteriori informazioni sulle caratteristiche e le funzionalità supportate dalla raccolta di messaggistica Amazon SQS Java, consulta le domande frequenti su Amazon SQS.

Interfacce comuni supportate

- `Connection`
- `ConnectionFactory`
- `Destination`
- `Session`
- `MessageConsumer`
- `MessageProducer`

Tipi di messaggi supportati

- `ByteMessage`

- `ObjectMessage`
- `TextMessage`

Modalità di riconoscimento del messaggio supportate

- `AUTO_ACKNOWLEDGE`
- `CLIENT_ACKNOWLEDGE`
- `DUPS_OK_ACKNOWLEDGE`
- `UNORDERED_ACKNOWLEDGE`

> **Note**
>
> La modalità `UNORDERED_ACKNOWLEDGE` non fa parte della specifica JMS 1.1. Grazie a questa modalità, Amazon SQS può consentire a un client JMS di dichiarare esplicitamente un messaggio.

Intestazioni definite da JMS e proprietà riservate

Per l'invio di messaggi

Quando invii messaggi, puoi impostare le seguenti intestazioni e proprietà per ogni messaggio:

- `JMSXGroupID` (necessaria per le code FIFO, non consentita per le code standard)
- `JMS_SQS_DeduplicationId` (opzionale per le code FIFO, non consentita per le code standard)

Dopo che invii messaggi, Amazon SQS imposta le seguenti intestazioni e proprietà per ogni messaggio:

- `JMSMessageID`
- `JMS_SQS_SequenceNumber` (solo per code FIFO)

Per la ricezione di messaggi

Quando ricevi messaggi, Amazon SQS imposta le seguenti intestazioni e proprietà per ogni messaggio:

- `JMSDestination`
- `JMSMessageID`
- `JMSRedelivered`
- `JMSXDeliveryCount`
- `JMSXGroupID` (solo per code FIFO)
- `JMS_SQS_DeduplicationId` (solo per code FIFO)
- `JMS_SQS_SequenceNumber` (solo per code FIFO)

Tutorial per Amazon SQS

Questa sezione fornisce tutorial che puoi utilizzare per esplorare caratteristiche e funzionalità di Amazon SQS.

Argomenti

Creazione di una coda Amazon SQS (AWS CloudFormation)

Puoi utilizzare la console AWS CloudFormation e un modello JSON (o YAML) per creare una coda Amazon SQS. Per ulteriori informazioni, consulta Lavorare con i modelli AWS CloudFormation e Risorse `AWS::SQS::Queue` nella Guida per gli utenti di AWS CloudFormation.

Usare AWS CloudFormation per la creazione di una coda Amazon SQS

1. Copia il seguente codice JSON in un file denominato `MyQueue.json`. Per creare una coda standard, ometti le proprietà `FifoQueue` e `ContentBasedDeduplication`. Per ulteriori informazioni sulla deduplicazione basata sui contenuti, consulta Elaborazione "exactly-once".

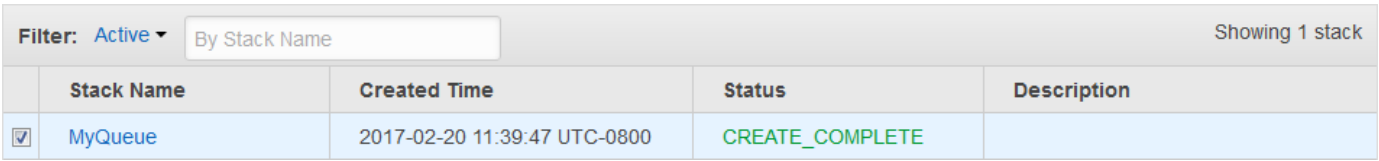
   > **Note**
   >
   > Il nome di una coda FIFO deve terminare con il suffisso `.fifo`.

   ```
   {
      "AWSTemplateFormatVersion": "2010-09-09",
      "Resources": {
         "MyQueue": {
            "Properties": {
               "QueueName": "MyQueue.fifo",
               "FifoQueue": true,
               "ContentBasedDeduplication": true
                },
            "Type": "AWS::SQS::Queue"
            }
         },
   ```

```
   "Outputs": {
      "QueueName": {
         "Description": "The name of the queue",
         "Value": {
            "Fn::GetAtt": [
               "MyQueue",
               "QueueName"
            ]
         }
      },
      "QueueURL": {
         "Description": "The URL of the queue",
         "Value": {
            "Ref": "MyQueue"
         }
      },
      "QueueARN": {
         "Description": "The ARN of the queue",
         "Value": {
            "Fn::GetAtt": [
               "MyQueue",
               "Arn"
            ]
         }
      }
   }
}
```

2. Accedi alla console AWS CloudFormation, quindi scegli Create Stack (Crea stack).
3. Nel pannello Specify Template (Specifica modello) scegliere Upload a template file (Carica file modello), scegliere il file `MyQueue.json`, quindi scegliere Next (Successivo).
4. Nella pagina Specify Details (Specifica dettagli), digita `MyQueue` per Stack Name (Nome stack), quindi scegli Next (Avanti).
5. Nella pagina Opzioni, scegli Next (Avanti).
6. Nella pagina Revisione scegli Create (Crea).

   AWS CloudFormation inizia a creare lo stack `MyQueue` e visualizza lo stato CREATE_IN_PROGRESS. Al termine del processo, AWS CloudFormation mostra lo stato CREATE_COMPLETE.



7. (Facoltativo) Per visualizzare il nome, l'URL e l'ARN della coda, scegli il nome dello stack e quindi nella pagina successiva espandi la sezione Outputs.

Tutorial: Invio di un messaggio a una coda Amazon SQS da Amazon Virtual Private Cloud

In questo tutorial viene illustrato come inviare messaggi a una coda Amazon SQS utilizzando una rete privata sicura. Questa rete è costituita da un VPC che contiene un'istanza Amazon EC2. L'istanza si connette a Amazon SQS tramite un endpoint VPC dell'interfaccia, consentendoti di effettuare la connessione all'istanza Amazon EC2 e di inviare messaggi alla coda Amazon SQS anche senza connessione alla rete Internet pubblica. Per ulteriori informazioni, consulta Endpoint di Amazon Virtual Private Cloud per Amazon SQS.

Important

- Puoi utilizzare Amazon Virtual Private Cloud solo con endpoint Amazon SQS HTTPS.
- Quando configuri Amazon SQS per inviare messaggi da Amazon VPC, devi abilitare il DNS privato e specificare gli endpoint nel formato `sqs.us-east-2.amazonaws.com`.
- Il DNS privato non supporta endpoint precedenti come `queue.amazonaws.com` o `us-east-2.queue.amazonaws.com`.

Argomenti

Fase 1: Creazione di una coppia di chiavi di Amazon EC2

Una coppia di chiavi consente di effettuare la connessione a un'istanza Amazon EC2. È costituita da una chiave pubblica per la crittografia delle informazioni di accesso e una chiave privata per decrittografare tali informazioni.

1. Accedi alla console Amazon EC2.
2. Nel menu di navigazione, in Network & Security (Rete e sicurezza), scegliere Key Pairs (Coppie di chiavi).
3. Scegli Crea coppia di chiavi.
4. Nella finestra di dialogo Create Key Pair (Crea coppia di chiavi), per Key pair name (Nome coppia di chiavi), immettere `SQS-VPCE-Tutorial-Key-Pair`, quindi scegliere Create (Crea).
5. Il browser scarica automaticamente il file della chiave privata `SQS-VPCE-Tutorial-Key-Pair.pem`.

   > **Important**
   >
   > Salvare il file in un percorso sicuro. EC2 non genera un secondo file `.pem` per la stessa coppia di chiavi.

6. Per consentire a un client SSH di connettersi all'istanza EC2, impostare le autorizzazioni per il file della chiave privata in modo che solo l'utente specificato abbia le relative autorizzazioni in lettura. Ad esempio:

   ```
   chmod 400 SQS-VPCE-Tutorial-Key-Pair.pem
   ```

Fase 2: creazione delle risorse AWS

Per configurare l'infrastruttura necessaria, devi utilizzare un AWS CloudFormation modello, che è un modello per creare uno stack composto da AWS risorse, come istanze Amazon EC2 e code Amazon SQS.

Lo stack per questo tutorial include le risorse seguenti:

- Un VPC e le risorse di rete associate, tra cui una sottorete, un gruppo di sicurezza, un gateway Internet e una tabella di routing

- Un'istanza Amazon EC2 avviata nella sottorete VPC
- Una coda Amazon SQS

1. Scarica il AWS CloudFormation modello denominato SQS-VPCE-Tutorial-CloudFormation.yamldaGitHub.
2. Accedere alla console AWS CloudFormation.
3. Scegli Create Stack (Crea stack).
4. Nella pagina Select Template (Seleziona modello), scegliere Upload a template to Amazon S3 (Carica un modello in Amazon S3), selezionare il file `SQS-VPCE-SQS-Tutorial-CloudFormation.yaml`, quindi scegliere Next (Successivo).
5. Nella pagina Specify Details (Specifica dettagli), procedi come segue:
   a. In Nome stack, immetti `SQS-VPCE-Tutorial-Stack`.
   b. Per KeyName, scegliete SQS-VPCE-Tutorial-Key-Pair.
   c. Seleziona Avanti.
6. Nella pagina Opzioni, scegli Avanti.
7. Nella pagina Revisione, nella sezione Funzionalità, scegli Riconosco che potrebbe creare risorse IAM con nomi personalizzati. AWS CloudFormation , quindi scegli Crea.

AWS CloudFormation inizia a creare lo stack e visualizza lo stato CREATE_IN_PROGRESS. Al termine del processo, AWS CloudFormation mostra lo stato CREATE_COMPLETE.

Fase 3: Verifica del fatto che l'istanza EC2 non è accessibile pubblicamente

Il modello AWS CloudFormation avvia l'istanza EC2 `SQS-VPCE-Tutorial-EC2-Instance` nel VPC. Tale istanza EC2 non consente il traffico in uscita e non è in grado di inviare messaggi a Amazon SQS. Per verificare ciò, devi connetterti all'istanza, provare a effettuare una connessione a un endpoint pubblico e tentare di inviare un messaggio a Amazon SQS.

1. Accedi alla console Amazon EC2.
2. Nel menu di navigazione, in Istanze, scegli Istanze.
3. Selezionare SQS-VPCE-Tutorial-EC2Instance.
4. Copiare il nome host visualizzato in Public DNS (IPv4) (DNS pubblico (IPv4)), ad esempio ec2-203-0-113-0.us-west-2.compute.amazonaws.com.

5. Dalla directory contenente la coppia di chiavi creata in precedenza, effettuare la connessione all'istanza con il seguente comando. Ad esempio:

```
ssh -i SQS-VPCE-Tutorial-Key-Pair.pem ec2-user@ec2-203-0-113-0.us-east-2.compute.amazonaws.com
```

6. Provare a effettuare la connessione a un qualsiasi endpoint pubblico. Ad esempio:

```
ping amazon.com
```

   Il tentativo di connessione avrà esito negativo, come previsto.

7. Accedere alla console Amazon SQS.
8. Dall'elenco delle code, selezionare la coda creata dal modello AWS CloudFormation. Ad esempio, VPCE-SQS-Tutorial-Stack-CFQueue-1ABCDEFGH2IJK.
9. Nella tabella Dettagli, copiare l'URL, ad esempio, https://sqs.us-east-2.amazonaws.com/123456789012/.
10. Dall'istanza EC2, provare a pubblicare un messaggio nella coda utilizzando il seguente comando. Ad esempio:

```
aws sqs send-message --region us-east-2 --endpoint-url https://sqs.us-east-2.amazonaws.com/ --queue-url https://sqs.us-east-2.amazonaws.com/123456789012/
 --message-body "Hello from Amazon SQS."
```

    Il tentativo di invio avrà esito negativo, come previsto.

> **Important**
>
> Successivamente, una volta creato un endpoint VPC per Aamzon SQS, il tentativo di invio andrà a buon fine.

Fase 4: Creazione di un endpoint Amazon VPC per Amazon SQS

Per collegare il tuo VPC a Amazon SQS, definisci un endpoint VPC dell'interfaccia. Dopo avere aggiunto l'endpoint, sarà possibile utilizzare l'API Amazon SQS dall'istanza EC2 nel VPC. Ciò consente di inviare messaggi a una coda della rete AWS senza utilizzare la rete Internet pubblica.

> **Note**
>
> L'istanza EC2 non dispone ancora dell'accesso ad altri endpoint e servizi AWS su Internet.

1. Accedere alla console Amazon VPC.
2. Nel menu di navigazione, scegliere Endpoints (Endpoint).
3. Scegliere Create Endpoint (Crea endpoint).
4. Nella pagina Crea endpoint, per Nome servizio, scegliere il nome del servizio per Amazon SQS.

   > **Note**
   >
   > I nomi dei servizi variano in base alla regione AWS corrente. Ad esempio, negli Stati Uniti orientali (Ohio), il nome del srevizio è com.amazonaws.*us-east-2*.sqs.

5. Per VPC, scegliere SQS-VPCE-Tutorial-VPC.
6. Per Subnets (Sottoreti), scegliere la sottorete il cui Subnet ID (ID sottorete) include SQS-VPCE-Tutorial-Subnet.
7. Per Security group (Gruppo di sicurezza), scegliere Select security groups (Seleziona gruppi di sicurezza), quindi scegliere il gruppo di sicurezza il cui Group Name (Nome gruppo) include SQS VPCE Tutorial Security Group (Gruppo di sicurezza tutorial SQS VPCE).
8. Seleziona Crea endpoint.

   Viene creato l'endpoint VPC e ne viene visualizzato il relativo ID. Ad esempio, vpce-0ab1cdef2ghi3j456k.
9. Scegli Chiudi.

   La console Amazon VPC apre la pagina Endpoints.

Amazon VPC inizia a creare l'endpoint e visualizza lo stato in attesa. Al termine del processo, Amazon VPC visualizza lo stato disponibile.

Fase 5: Invio di un messaggio alla coda Amazon SQS

Il VPC include un endpoint per Amazon SQS. Puoi connetterti all'istanza EC2 e inviare messaggi alla coda.

1. Effettuare nuovamente la connessione all'istanza EC2. Ad esempio:

```
ssh -i SQS-VPCE-Tutorial-Key-Pair.pem ec2-user@ec2-203-0-113-0.us-
east-2.compute.amazonaws.com
```

2. Provare a pubblicare nuovamente un messaggio nella coda utilizzando il seguente comando. Ad esempio:

```
aws sqs send-message --region us-east-2 --endpoint-url https://sqs.us-
east-2.amazonaws.com/ --queue-url https://sqs.us-east-2.amazonaws.com/123456789012/
 --message-body "Hello from Amazon SQS."
```

   Il tentativo di invio andrà a buon fine e verranno visualizzati il digest MD5 del corpo del messaggio e l'ID messaggio.

```
{
 "MD5OfMessageBody": "a1bcd2ef3g45hi678j90klmn12p34qr5",
 "MessageId": "12345a67-8901-2345-bc67-d890123e45fg"
}
```

Per informazioni sulla ricezione e l'eliminazione del messaggio dalla coda creata dal modello AWS CloudFormation (ad esempio, VPCE-SQS-Tutorial-Stack-CFQueue-1ABCDEFGH2IJK), consulta Ricevere ed eliminare un messaggio (console).

Per informazioni sull'eliminazione delle risorse, consulta quanto segue:

- Eliminazione di un endpoint VPC nella Guida per l'utente di Amazon VPC
- Elimina una coda
- Termina l'istanza nella Guida per l'utente di Amazon EC2 per le istanze Linux
- Eliminazione di un VPC nella Guida per l'utente di Amazon VPC
- Eliminazione di uno stack nella console AWS CloudFormation nella Guida per l'utente di AWS CloudFormation
- Eliminazione della coppia di chiavi nella Guida per l'utente di Amazon EC2 per le istanze Linux

Automazione e risoluzione dei problemi delle code di Amazon SQS

Questa sezione fornisce informazioni sull'automazione e la risoluzione dei problemi delle code di Amazon SQS.

Argomenti

Automazione delle notifiche dai servizi AWS ad Amazon SQS con Amazon EventBridge

Amazon EventBridge consente di automatizzare i servizi AWS e di rispondere a eventi di sistema, come i problemi relativi alla disponibilità delle applicazioni o alle modifiche della risorsa. Gli eventi dei servizi AWS vengono recapitati a EventBridge quasi in tempo reale. Puoi compilare regole semplici che indichino quali eventi sono considerati di interesse per te e quali operazioni automatizzate intraprendere quando un evento corrisponde a una regola.

EventBridge ti consente di impostare un'ampia gamma di destinazioni, ad esempio Amazon SQS standard e le code FIFO, che ricevono gli eventi in formato JSON. Per ulteriori informazioni, consulta Obiettivi Amazon EventBridge nella Guida per l'utente di Amazon EventBridge.

Risoluzione dei problemi delle code Amazon Simple Queue Service utilizzando AWS X-Ray

AWS X-Ray raccoglie i dati sulle richieste relative all'applicazione e ti consente di visualizzare e filtrare i dati per identificare i potenziali problemi e le opportunità di ottimizzazione. Per qualsiasi richiesta tracciata che raggiunge la tua applicazione, puoi visualizzare informazioni dettagliate non solo sulla richiesta ma anche sulle chiamate che la tua applicazione esegue verso le risorse AWS, i microservizi, i database e le API web HTTP a valle.

Per inviare intestazioni di traccia AWS X-Ray tramite Amazon SQS, puoi effettuare una delle seguenti operazioni:

- Usare l'intestazione di traccia `X-Amzn-Trace-Id`.
- Usare gli attributi del sistema di messaggi `AWSTraceHeader`.

Per raccogliere dati relativi a errori e latenza, è necessario analizzare il client `AmazonSQS` utilizzando l'AWSSDK X-Ray.

È possibile utilizzare la console AWS X-Ray per visualizzare la mappa delle connessioni tra Amazon SQS e altri servizi impiegati dall'applicazione. È inoltre possibile utilizzare la console per visualizzare i parametri come la latenza media e le percentuali di errore. Per ulteriori informazioni, consultare Amazon SQS e AWS X-Ray nella Guida per sviluppatori di AWS X-Ray.

Sicurezza in Amazon SQS

Questa sezione fornisce informazioni sulla sicurezza di Amazon SNS, l'autenticazione e il controllo degli accessi, oltre alla sintassi delle policy di accesso Amazon SQS.

Argomenti

Protezione dei dati

Il modello di responsabilità AWS condivisa modello si applica alla protezione dei dati in Amazon Simple Queue Service. Come descritto in questo modello, AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutti i Cloud AWS. L'utente è responsabile del controllo dei contenuti ospitati su questa infrastruttura. Inoltre, sei responsabile della configurazione della protezione e delle attività di gestione per i Servizi AWS che utilizzi. Per ulteriori informazioni sulla privacy dei dati, vedi Domande frequenti sulla privacy dei dati. Per informazioni sulla protezione dei dati in Europa, consulta il post del blog AWS Shared Responsibility Model and GDPR nel Blog sulla sicurezza AWS .

Ai fini della protezione dei dati, consigliamo di proteggere Account AWS le credenziali e configurare i singoli utenti con AWS IAM Identity Center or AWS Identity and Access Management (IAM). In tal modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere i suoi compiti. Ti suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Utilizza l'autenticazione a più fattori (MFA) con ogni account.
- Usa SSL/TLS per comunicare con le risorse. AWS È richiesto TLS 1.2 ed è consigliato TLS 1.3.
- Configura l'API e la registrazione delle attività degli utenti con. AWS CloudTrail
- Utilizza soluzioni di AWS crittografia, insieme a tutti i controlli di sicurezza predefiniti all'interno Servizi AWS.

- Utilizza i servizi di sicurezza gestiti avanzati, come Amazon Macie, che aiutano a individuare e proteggere i dati sensibili archiviati in Amazon S3.
- Se hai bisogno di moduli crittografici convalidati FIPS 140-2 per l'accesso AWS tramite un'interfaccia a riga di comando o un'API, utilizza un endpoint FIPS. Per ulteriori informazioni sugli endpoint FIPS disponibili, consulta il Federal Information Processing Standard (FIPS) 140-2.

Ti consigliamo vivamente di non inserire mai informazioni riservate o sensibili, ad esempio gli indirizzi e-mail dei clienti, nei tag o nei campi di testo in formato libero, ad esempio nel campo Nome. Ciò include quando lavori con Amazon SQS o altro Servizi AWS utilizzando la console, l'API o AWS gli AWS CLI SDK. I dati inseriti nei tag o nei campi di testo in formato libero utilizzati per i nomi possono essere utilizzati per i la fatturazione o i log di diagnostica. Quando fornisci un URL a un server esterno, ti suggeriamo vivamente di non includere informazioni sulle credenziali nell'URL per convalidare la tua richiesta al server.

Le sezioni seguenti forniscono informazioni sulla protezione dei dati in Amazon SQS.

Argomenti

Crittografia dei dati

La protezione dei dati ha lo scopo di proteggere i dati sia in transito (durante la trasmissione verso e da Amazon SQS), sia quando sono inattivi (ovvero quando sono archiviati su disco nei data center Amazon SQS). È possibile proteggere i dati in transito tramite il protocollo Secure Sockets Layer (SSL) o tramite la crittografia lato client. Per impostazione predefinita, Amazon SQS archivia messaggi e file utilizzando la crittografia del disco. Puoi proteggere i dati inattivi richiedendo ad Amazon SQS di crittografare i tuoi messaggi prima di salvarli nel file system crittografato dei suoi data center. Amazon SQS consiglia di utilizzare SSE per la crittografia ottimizzata dei dati.

Argomenti

Crittografia a riposo

La crittografia lato server (SSE) consente di trasmettere dati sensibili in code crittografate. SSE protegge il contenuto dei messaggi nelle code utilizzando chiavi di crittografia gestite da SQL (SSE-SQS) o chiavi gestite in (SSE-KMS). AWS Key Management Service Per informazioni sulla gestione di SSE tramite, consulta quanto segue: AWS Management Console

- Configurare SSE-SQS per una coda (console)
- Configurare SSE-KMS per una coda (console)

Per informazioni sulla gestione di SSE utilizzando le `GetQueueAttributes` azioni AWS SDK for Java (e `CreateQueueSetQueueAttributes`, e), consulta i seguenti esempi:

- Utilizzo della crittografia lato server (SSE)
- Configurazione delle autorizzazioni KMS per Servizi AWS

SSE esegue la crittografia dei messaggi non appena vengono ricevuti da Amazon SQS. I messaggi sono archiviati in forma crittografata e decrittografati da Amazon SQS solo quando vengono inviati a un consumatore autorizzato.

> **Important**
>
> Tutte le richieste alle code con la funzione SSE abilitata devono utilizzare HTTPS e Signature Version 4.
> Una coda crittografata che utilizza la chiave predefinita (chiave KMS AWS gestita per Amazon SQS) non può richiamare una funzione Lambda in un'altra. Account AWS
> Alcune funzionalità dei AWS servizi che possono inviare notifiche ad Amazon SQS utilizzando l' AWS Security Token Service `AssumeRole`azione sono compatibili con SSE ma funzionano solo con le code standard:
>
> - Hook del ciclo di vita di dimensionamento automatico
> - AWS Lambda Code DLQ
>
> Per informazioni sulla compatibilità di altri servizi con code crittografate, consulta AWS Configura le autorizzazioni KMS per i servizi e la documentazione del servizio.

AWS KMS combina hardware e software sicuri e ad alta disponibilità per fornire un sistema di gestione delle chiavi scalabile per il cloud. Quando usi Amazon SQS con AWS KMS, anche le chiavi dati che crittografano i dati dei messaggi vengono crittografate e archiviate con i dati che proteggono.

Di seguito sono elencati i vantaggi derivanti dall'uso di AWS KMS:

- È possibile creare e gestire AWS KMS keys in modo autonomo.
- Puoi anche utilizzare la chiave KMS AWS gestita per Amazon SQS, che è unica per ogni account e regione.
- Gli standard AWS KMS di sicurezza possono aiutarti a soddisfare i requisiti di conformità relativi alla crittografia.

Per ulteriori informazioni, consulta Cos'è AWS Key Management Service? nella Guida per gli sviluppatori AWS Key Management Service .

Argomenti

- Ambito della crittografia
- Termini chiave

Ambito della crittografia

SSE crittografa il corpo di un messaggio in una coda Amazon SQS.

SSE non esegue la crittografia di quanto segue:

- Metadati della coda (nome e attributi della coda)
- Metadati del messaggio (ID messaggio, timestamp e attributi)
- Parametri per coda

La crittografia di un messaggio ne rende i contenuti non disponibili a utenti non autorizzati o anonimi. Con SSE abilitato, le richieste anonime `SendMessage` e `ReceiveMessage` alla coda crittografata verranno rifiutate. Le best practice di sicurezza di Amazon SQS consigliano di non utilizzare richieste anonime. Se desideri inviare richieste anonime a una coda Amazon SQS, assicurati di disabilitare SSE. Ciò non ha implicazioni sul normale funzionamento di Amazon SQS:

- Un messaggio viene crittografato solo se inviato dopo che la crittografia di una coda è abilitata. Amazon SQS non crittografa i messaggi in backlog.

- Tutti i messaggi crittografati restano crittografati anche se la crittografia della relativa coda è disabilitata.

Lo spostamento di un messaggio a una coda dead-letter non ne pregiudica la crittografia:

- Quando Amazon SQS sposta un messaggio da una coda di origine crittografata a una coda DLQ non crittografata, il messaggio rimane crittografato.
- Quando Amazon SQS sposta un messaggio da una coda di origine non crittografata a una coda DLQ crittografata, il messaggio rimane non crittografato.

Termini chiave

I seguenti termini chiave possono aiutarti a comprendere meglio le funzionalità di SSE . Per una descrizione dettagliata, consulta la Documentazione di riferimento per l'API Amazon Simple Notification Service.

Chiave di dati

La chiave (DEK) responsabile della crittografia dei contenuti dei messaggi Amazon SQS.

Per ulteriori informazioni, consulta Chiavi di dati nella Guida per sviluppatori di AWS Key Management Service nella Guida per sviluppatori di AWS Encryption SDK .

Periodo di riutilizzo della chiave di dati

Il periodo di tempo, in secondi, durante il quale Amazon SQS può riutilizzare una chiave dati per crittografare o decrittografare i messaggi prima di effettuare una nuova chiamata. AWS KMS Numero intero che rappresenta secondi, tra 60 (1 minuto) e 86.400 (24 ore). Il valore predefinito è 300 (5 minuti). Per ulteriori informazioni, consulta Informazioni sul periodo di riutilizzo della chiave di dati.

> **Note**
>
> Nell'improbabile eventualità di impossibilità di raggiungerla AWS KMS, Amazon SQS continua a utilizzare la chiave dati memorizzata nella cache fino a quando non viene ristabilita una connessione.

ID della chiave KMS

L'alias, l'alias ARN, l'ID della chiave o l'ARN della chiave KMS AWS gestita o di una chiave KMS personalizzata, nel tuo account o in un altro account. Sebbene l'alias della chiave KMS AWS gestita per Amazon SQS sia `alias/aws/sqs` sempre, l'alias di una chiave KMS personalizzata può, ad esempio, essere. `alias/MyAlias` Puoi utilizzare queste chiavi KMS per proteggere i messaggi nelle code Amazon SQS.

> **Note**
>
> Ricorda quanto segue:
>
> - Se non specifichi una chiave KMS personalizzata, Amazon SQS utilizza AWS la chiave KMS gestita per Amazon SQS.
> - La prima volta che utilizzi AWS Management Console per specificare la chiave KMS AWS gestita per Amazon SQS per una coda AWS KMS , crea AWS la chiave KMS gestita per Amazon SQS.
> - In alternativa, la prima volta che utilizzi l'`SendMessageBatch`azione `SendMessage` o su una coda con SSE abilitato, AWS KMS crea la chiave KMS AWS gestita per Amazon SQS.

Puoi creare chiavi KMS, definire le politiche che controllano l'utilizzo delle chiavi KMS e controllare l'utilizzo delle chiavi KMS utilizzando la sezione Customer managed keys della console o dell'azione. AWS KMS `CreateKey` AWS KMS Per ulteriori informazioni, consulta Chiavi KMS e Creazione di chiavi nella Guida per gli sviluppatori di AWS Key Management Service . Per altri esempi di identificatori di chiave KMS, consulta KeyIdl'API Reference. AWS Key Management Service Per ulteriori informazioni su come individuare gli identificatori KMS, consulta Trovare l'ARN e l'ID chiave nella Guida per sviluppatori di AWS Key Management Service .

> **Important**
>
> Sono previsti costi aggiuntivi per l'utilizzo. AWS KMS Per ulteriori informazioni, consulta Stima dei costi AWS KMS e Prezzi di AWS Key Management Service.

Crittografia a busta

La sicurezza dei dati crittografati dipende in parte dalla protezione della chiave di dati che può decrittarli. Amazon SQS utilizza la chiave KMS per crittografare la chiave di dati, quindi la chiave di dati crittografata viene archiviata con il messaggio crittografato. Questa pratica di utilizzare una chiave KMS per crittografare le chiavi di dati è nota come crittografia a busta.

Per ulteriori informazioni, consulta Crittografia a busta nella Guida per gli sviluppatori di AWS Encryption SDK .

Gestione delle chiavi

Amazon SQS si integra con AWS Key Management Service (KMS) per gestire le chiavi KMS per la crittografia lato server (SSE). Per informazioni su SSE e definizioni di gestione delle chiavi, consulta la sezione Crittografia a riposo. Amazon SQS utilizza le chiavi KMS per convalidare e proteggere le chiavi di dati che crittografano e decrittografano i messaggi. Nelle sezioni seguenti vengono fornite informazioni sull'utilizzo delle chiavi KMS e delle chiavi di dati nel servizio Amazon SQS.

Argomenti

- Configurazione delle autorizzazioni per AWS KMS
- Informazioni sul periodo di riutilizzo della chiave di dati
- Stima dei costi AWS KMS
- AWS KMS errori

Configurazione delle autorizzazioni per AWS KMS

Ogni chiave KMS deve avere una policy delle chiavi. Tieni presente che non puoi modificare la politica delle chiavi di una chiave KMS AWS gestita per Amazon SQS. La policy per questa chiave KMS include le autorizzazioni per tutte le entità principali nell'account (che sono autorizzate a utilizzare Amazon SQS) per usare le code crittografate.

Per una KMS gestita dal cliente, è necessario configurare la policy della chiave per aggiungere autorizzazioni per ogni produttore e consumatore della coda. A tale scopo, nomini il produttore e il consumatore come utenti nella policy della chiave KMS. Per ulteriori informazioni sulle AWS KMS autorizzazioni, consulta le AWS KMS risorse e le operazioni o il riferimento alle autorizzazioni AWS KMS API nella Guida per gli sviluppatori. AWS Key Management Service

In alternativa, puoi specificare le autorizzazioni richieste in una policy IAM assegnata alle entità principal che producono e utilizzano messaggi crittografati. Per ulteriori informazioni, consulta Utilizzo delle policy IAM con AWS KMS nella Guida per gli sviluppatori di AWS Key Management Service .

> **Note**
>
> Sebbene sia possibile configurare le autorizzazioni globali per l'invio e la ricezione da Amazon SQS AWS KMS , è necessario denominare in modo esplicito l'ARN completo delle chiavi KMS in regioni `Resource` specifiche nella sezione di una policy IAM.

AWS Configura le autorizzazioni KMS per i servizi

Diversi AWS servizi fungono da sorgenti di eventi in grado di inviare eventi alle code di Amazon SQS. Per consentire a queste fonti di eventi di funzionare con code crittografate, devi creare una chiave KMS gestita dal cliente e aggiungere le autorizzazioni nella policy chiave affinché il servizio utilizzi i metodi API richiesti. AWS KMS Esegui la procedura seguente per configurare le autorizzazioni.

1. Crea una chiave KMS gestita dal cliente Per ulteriori informazioni, consulta Creazione di chiavi nella Guida per gli sviluppatori di AWS Key Management Service .
2. Per consentire all'origine degli eventi del AWS servizio di utilizzare i metodi `kms:GenerateDataKey` e `kms:Decrypt` API, aggiungi la seguente dichiarazione alla politica delle chiavi KMS.

```
{
   "Version": "2012-10-17",
      "Statement": [{
         "Effect": "Allow",
         "Principal": {
            "Service": "service.amazonaws.com"
         },
         "Action": [
            "kms:GenerateDataKey",
            "kms:Decrypt"
         ],
         "Resource": "*"
       }]
}
```

Sostituire "service" nell'esempio precedente con il nome del servizio dell'origine evento. Le origini eventi includono i seguenti servizi.

Origine eventi	Nome servizio
CloudWatch Eventi Amazon	`events.amazonaws.com`
Notifiche di eventi Amazon S3	`s3.amazonaws.com`
Sottoscrizioni ad argomenti Amazon SNS	`sns.amazonaws.com`

3. Configurare una coda SSE esistente utilizzando l'ARN della tua chiave KMS.
4. Fornire l'ARN della coda crittografata per l'origine eventi.

Configurare le autorizzazioni KMS per i produttori

Quando scade il periodo di riutilizzo della chiave dati, la successiva chiamata del produttore a `SendMessage` o `SendMessageBatch` attiva anche le chiamate a `kms:GenerateDataKey` e `kms:Decrypt`. La chiamata a `kms:Decrypt` è per verificare l'integrità della nuova chiave dati prima di utilizzarla. Il produttore deve quindi avere le autorizzazioni `kms:GenerateDataKey` e `kms:Decrypt` per la chiave KMS.

Aggiungere la seguente istruzione alla policy IAM del produttore. Ricordarsi di utilizzare i valori ARN corretti per la risorsa chiave e la risorsa coda.

```
{
   "Version": "2012-10-17",
      "Statement": [{
         "Effect": "Allow",
         "Action": [
            "kms:GenerateDataKey",
            "kms:Decrypt"
         ],
         "Resource":  "arn:aws:kms:us-
east-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
         }, {
         "Effect": "Allow",
         "Action": [
            "sqs:SendMessage"
         ],
```

```
         "Resource": "arn:aws:sqs:*:123456789012:MyQueue"
      }]
}
```

Configurare le autorizzazioni KMS per i consumatori

Quando scade il periodo di riutilizzo della chiave dati, la successiva chiamata del consumatore a `ReceiveMessage` attiva anche una chiamata a `kms:Decrypt` per verificare l'integrità della nuova chiave dati prima di utilizzarla. Il consumatore deve avere l'autorizzazione `kms:Decrypt` per qualsiasi chiave KMS che viene utilizzata per crittografare i messaggi nella coda specificata. Se una coda funge da coda DLQ, il consumatore deve avere anche l'autorizzazione `kms:Decrypt` per qualsiasi chiave KMS che viene utilizzata per crittografare i messaggi nella coda di origine. Aggiungere la seguente istruzione alla policy IAM del consumatore. Ricordarsi di utilizzare i valori ARN corretti per la risorsa chiave e la risorsa coda.

```
{
   "Version": "2012-10-17",
      "Statement": [{
         "Effect": "Allow",
         "Action": [
            "kms:Decrypt"
         ],
         "Resource": "arn:aws:kms:us-
east-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
         }, {
         "Effect": "Allow",
         "Action": [
            "sqs:ReceiveMessage"
         ],
         "Resource": "arn:aws:sqs:*:123456789012:MyQueue"
      }]
}
```

Configurare le autorizzazioni KMS con protezione dal "confuse deputy"

Quando il principale di una istruzione della policy della chiave è un Principale del servizio AWS, è possibile utilizzare le chiavi di condizione globali `aws:SourceArn` o `aws:SourceAccount` per proteggersi dal problema del "confused deputy". Per utilizzare queste chiavi di condizione, impostare il valore sul nome della risorsa Amazon (ARN) della risorsa crittografata. Se non si conosce l'ARN della risorsa, utilizzare `aws:SourceAccount`.

In questa policy della chiave KMS, una risorsa specifica del servizio di proprietà dell'account `111122223333` può richiamare KMS per le operazioni `Decrypt` e `GenerateDataKey` che si verificano durante l'utilizzo SSE di Amazon SQS.

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Principal": {
            "Service": "<replaceable>service</replaceable>.amazonaws.com"
  },
  "Action": [
   "kms:GenerateDataKey",
   "kms:Decrypt"
  ],
  "Resource": "*",
  "Condition": {
   "ArnEquals": {
    "aws:SourceArn": [
     "arn:aws:service::111122223333:resource"
    ]
   }
  }
 }]
}
```

Quando si utilizzano code Amazon SQS abilitate per SSE, sono supportati i seguenti servizi `aws:SourceArn`:

- Amazon SNS
- Amazon S3
- CloudWatch Eventi
- AWS Lambda
- CodeBuild
- Customer Profiles Amazon Connect
- AWS Auto Scaling
- Amazon Chime

Informazioni sul periodo di riutilizzo della chiave di dati

Il periodo di riutilizzo della chiave di dati definisce la durata massima per Amazon SQS per riutilizzare la stessa chiave dati. Quando termina il periodo di riutilizzo della chiave di dati, Amazon SQS genera una nuova chiave di dati. Prendere nota delle seguenti linee guida sul periodo di riutilizzo.

- Un periodo di riutilizzo più breve offre una maggiore sicurezza, ma comporta un maggior numero di chiamate verso AWS KMS, il che potrebbe comportare addebiti oltre il piano gratuito.
- Anche se la chiave di dati viene memorizzata nella cache separatamente per la crittografia e la decrittografia, il periodo di riutilizzo si applica a entrambe le copie della chiave di dati.
- Al termine del periodo di riutilizzo della chiave dati, la chiamata successiva `SendMessage` o `SendMessageBatch` in genere attiva una chiamata al AWS KMS `GenerateDataKey` metodo per ottenere una nuova chiave dati. Inoltre, ogni chiamata successiva a `SendMessage` e `ReceiveMessage` attiverà ciascuna una chiamata AWS KMS `Decrypt` a per verificare l'integrità della chiave dati prima di utilizzarla.
- I responsabili (Account AWS o gli utenti) non condividono le chiavi dati (i messaggi inviati da destinatari univoci ottengono sempre chiavi dati uniche). Pertanto, il volume delle chiamate verso AWS KMS è un multiplo del numero di principali univoci in uso durante il periodo di riutilizzo della chiave dati:

Stima dei costi AWS KMS

Per prevedere i costi e comprendere meglio la AWS fattura, potresti voler sapere con quale frequenza Amazon SQS utilizza la tua chiave KMS.

> **Note**
>
> Anche se la seguente formula può darti un'idea molto precisa dei costi previsti, i costi effettivi potrebbero essere più elevati a causa della natura diffusa di Amazon SQS.

Per calcolare il numero di richieste API (R) per coda, usa la formula seguente:

```
R = (B / D) * (2 * P + C)
```

B è il periodo di fatturazione (in secondi).

D è il periodo di riutilizzo della chiave di dati (in secondi).

`P` è il numero di principali produttori che effettuano invii alla coda Amazon SQS.

`C` è il numero di principali consumatori che ricevono dalla coda Amazon SQS.

> **Important**
>
> In generale, ai principali produttori viene addebitato un importo doppio rispetto ai principali consumatori. Per ulteriori informazioni, consulta Informazioni sul periodo di riutilizzo della chiave di dati.
> Se il produttore e l'utilizzatore hanno utenti diversi, il costo aumenta.

Di seguito vengono riportati esempi di calcolo. Per informazioni dettagliate sui prezzi, consulta Prezzi di AWS Key Management Service.

Esempio 1: calcolo del numero di chiamate AWS KMS API per 2 principali e 1 coda

Questo esempio assume quanto segue:

- Il periodo di fatturazione è compreso tra il 1° e il 31 gennaio (2.678.400 secondi).
- Il periodo di riutilizzo della chiave di dati è impostato su 5 minuti (300 secondi).
- C'è una coda.
- C'è 1 principale produttore e 1 principale consumatore.

```
(2,678,400 / 300) * (2 * 1 + 1) = 26,784
```

Esempio 2: calcolo del numero di chiamate AWS KMS API per più produttori e consumatori e 2 code

Questo esempio assume quanto segue:

- Il periodo di fatturazione è compreso tra il 1° e il 28 febbraio (2.419.200 secondi).
- Il periodo di riutilizzo della chiave di dati è impostato su 24 ore (86.400 secondi).
- Ci sono 2 code.
- La prima coda ha 3 principali produttori e 1 principale consumatore.
- La seconda coda ha 5 principali produttori e 2 principali consumatori.

```
(2,419,200 / 86,400 * (2 * 3 + 1)) + (2,419,200 / 86,400 * (2 * 5 + 2)) = 532
```

AWS KMS errori

Quando lavori con Amazon SQS e AWS KMS, potresti riscontrare errori. I seguenti riferimenti descrivono gli errori e le possibili soluzioni di risoluzione dei problemi.

- Errori comuni AWS KMS
- Errori di decrittografia AWS KMS
- AWS KMS GenerateDataKey errori

Riservatezza del traffico Internet

Un endpoint Amazon Virtual Private Cloud (Amazon VPC) per Amazon SQS è un'entità logica all'interno di un VPC che consente la connettività solo ad Amazon SQS. Il VPC instrada le richieste ad Amazon SQS e le risposte al VPC. Nelle sezioni seguenti vengono fornite informazioni sull'utilizzo degli endpoint VPC e sulla creazione delle policy di endpoint VPC.

Argomenti

- Endpoint di Amazon Virtual Private Cloud per Amazon SQS
- Creazione di una policy per endpoint VPC di Amazon per Amazon SQS

Endpoint di Amazon Virtual Private Cloud per Amazon SQS

Se utilizzi Amazon VPC per ospitare AWS le tue risorse, puoi stabilire una connessione tra il tuo VPC e Amazon SQS. Puoi utilizzare questa connessione per inviare messaggi alle code Amazon SQS senza utilizzare la rete Internet pubblica.

Amazon VPC ti consente di avviare AWS risorse in una rete virtuale personalizzata. Puoi utilizzare un VPC per controllare le impostazioni di rete, come l'intervallo di indirizzi IP, le sottoreti, le tabelle di routing e i gateway di rete. Per ulteriori informazioni sui VPC, consulta la Guida per l'utente di Amazon VPC.

Per connettere il tuo VPC a Amazon SQS, devi innanzitutto definire un'interfaccia dell'endpoint VPC, la quale ti consente di connettere il VPC ad altri servizi AWS . L'endpoint offre una connettività dimensionabile e affidabile a Amazon SQS senza richiedere un gateway internet, un'istanza NAT

(Network Address Translation) o una connessione VPN. Per ulteriori informazioni, consulta Tutorial: Invio di un messaggio a una coda Amazon SQS da Amazon Virtual Private Cloud e Esempio 5: Negare l'accesso se non è un endpoint VPC in questa guida ed Endpoint VPC dell'interfaccia (AWS PrivateLink) nella Guida per l'utente di Amazon VPC.

> **Important**
>
> - Puoi utilizzare Amazon Virtual Private Cloud solo con endpoint Amazon SQS HTTPS.
> - Quando configuri Amazon SQS per inviare messaggi da Amazon VPC, devi abilitare il DNS privato e specificare gli endpoint nel formato `sqs.us-east-2.amazonaws.com`.
> - Il DNS privato non supporta endpoint precedenti come `queue.amazonaws.com` o `us-east-2.queue.amazonaws.com`.

Creazione di una policy per endpoint VPC di Amazon per Amazon SQS

È possibile creare una policy per gli endpoint VPC di Amazon per Amazon SQS per specificare quanto segue:

- Il principale che può eseguire azioni.
- Le azioni che possono essere eseguite.
- Le risorse sui cui si possono eseguire azioni.

Per ulteriori informazioni, consulta Controllo degli accessi ai servizi con endpoint VPC nella Guida per l'utente di Amazon VPC.

Il seguente esempio di policy di endpoint VPC specifica che l'utente `MyUser` è autorizzato a inviare messaggi alla coda Amazon SQS `MyQueue`.

```
{
   "Statement": [{
      "Action": ["sqs:SendMessage"],
      "Effect": "Allow",
      "Resource": "arn:aws:sqs:us-east-2:123456789012:MyQueue",
      "Principal": {
        "AWS": "arn:aws:iam:123456789012:user/MyUser"
      }
   }]
```

```
}
```

Non si può accedere a quanto segue:

- Altre azioni API Amazon SQS, come `sqs:CreateQueue` e `sqs:DeleteQueue`.
- Altri utenti e ruoli che provano a utilizzare questo endpoint VPC.
- Invio di messaggi da `MyUser` a una coda Amazon SQS diversa.

> **Note**
>
> L'utente può ancora utilizzare altre azioni API Amazon SQS dall'esterno del VPC. Per ulteriori informazioni, consulta Esempio 5: Negare l'accesso se non è un endpoint VPC.

Identity and Access Management in Amazon SQS

AWS Identity and Access Management (IAM) è un Servizio AWSche consente agli amministratori di controllare in modo sicuro l'accesso alle risorse AWS. Gli amministratori IAM controllano chi può essere autenticato (accesso effettuato) e autorizzato (dispone di autorizzazioni) per utilizzare le risorse Amazon SQS. IAM è un Servizio AWS il cui uso non comporta costi aggiuntivi.

Destinatari

Le modalità di utilizzo di AWS Identity and Access Management (IAM) cambiano in base alle operazioni eseguite in Amazon SQS.

Utente del servizio: se utilizzi il servizio Amazon SQS per svolgere le tue mansioni, l'amministratore ti fornisce le credenziali e le autorizzazioni necessarie. All'aumentare del numero di funzionalità Amazon SQS utilizzate per svolgere il tuo lavoro, potrebbero rendersi necessarie ulteriori autorizzazioni. La comprensione della gestione dell'accesso ti consente di richiedere le autorizzazioni corrette all'amministratore. Se non riesci ad accedere a una funzionalità in Amazon SQS, consulta Risoluzione dei problemi di identità e accesso ad Amazon Simple Queue Service.

Amministratore del servizio: se sei responsabile delle risorse Amazon SQS presso la tua azienda, probabilmente disponi dell'accesso completo ad Amazon SQS. Il tuo compito è determinare le funzionalità e le risorse di Amazon SQS a cui gli utenti del servizio devono accedere. Devi inviare le richieste all'amministratore IAM per cambiare le autorizzazioni degli utenti del servizio. Esamina le informazioni contenute in questa pagina per comprendere i concetti di base relativi a IAM. Per ulteriori

informazioni su come la tua azienda può utilizzare IAM con Amazon SQ, consulta Come funziona Amazon Simple Queue Service con IAM.

Amministratore IAM: per gli amministratori IAM potrebbe essere vantaggioso ottenere informazioni su come scrivere policy per gestire l'accesso a Amazon SQS. Per visualizzare policy basate su identità Amazon SQS di esempio che possono essere utilizzate in IAM, consulta Best practice per le policy.

Autenticazione con identità

L'autenticazione è la procedura di accesso ad AWS con le credenziali di identità. Devi essere autenticato (connesso a AWS) come utente root Utente root dell'account AWS, come utente IAM o assumere un ruolo IAM.

Puoi accedere ad AWS come identità federata utilizzando le credenziali fornite attraverso un'origine di identità. Gli utenti AWS IAM Identity Center (Centro identità IAM), l'autenticazione Single Sign-On (SSO) dell'azienda e le credenziali di Google o Facebook sono esempi di identità federate. Se accedi come identità federata, l'amministratore ha configurato in precedenza la federazione delle identità utilizzando i ruoli IAM. Se accedi ad AWS tramite la federazione, assumi indirettamente un ruolo.

A seconda del tipo di utente, puoi accedere alla AWS Management Console o al portale di accesso AWS. Per ulteriori informazioni sull'accesso ad AWS, consulta la sezione Come accedere al tuo Account AWS nella Guida per l'utente di Accedi ad AWS.

Se accedi ad AWS in modo programmatico, AWS fornisce un Software Development Kit (SDK) e un'interfaccia a riga di comando (CLI) per firmare crittograficamente le richieste utilizzando le tue credenziali. Se non utilizzi gli strumenti AWS, devi firmare le richieste personalmente. Per ulteriori informazioni sulla firma delle richieste, consulta Firma delle richieste AWS nella Guida per l'utente IAM.

A prescindere dal metodo di autenticazione utilizzato, potrebbe essere necessario specificare ulteriori informazioni sulla sicurezza. AWS consiglia ad esempio di utilizzare l'autenticazione a più fattori (MFA) per aumentare la sicurezza dell'account. Per ulteriori informazioni, consulta Autenticazione a più fattori nella Guida per l'utente di AWS IAM Identity Center e Utilizzo dell'autenticazione a più fattori (MFA) in AWS nella Guida per l'utente di IAM.

Utente root di un Account AWS

Quando crei un Account AWS, inizi con una singola identità di accesso che ha accesso completo a tutti i Servizi AWS e le risorse nell'account. Tale identità è detta utente root Account AWS ed è possibile accedervi con l'indirizzo e-mail e la password utilizzati per creare l'account. Si consiglia

vivamente di non utilizzare l'utente root per le attività quotidiane. Conserva le credenziali dell'utente root e utilizzarle per eseguire le operazioni che solo l'utente root può eseguire. Per un elenco completo delle attività che richiedono l'accesso come utente root, consulta la sezione Attività che richiedono le credenziali dell'utente root nella Guida per l'utente di IAM.

Identità federata

Come best practice, richiedi agli utenti umani, compresi quelli che richiedono l'accesso di amministratore, di utilizzare la federazione con un provider di identità per accedere a Servizi AWSutilizzando credenziali temporanee.

Un'identità federata è un utente della directory degli utenti aziendali, un provider di identità Web, AWS Directory Service, la directory Identity Center o qualsiasi utente che accede ai Servizi AWS utilizzando le credenziali fornite tramite un'origine di identità. Quando le identità federate accedono agli Account AWS, assumono ruoli e i ruoli forniscono credenziali temporanee.

Per la gestione centralizzata degli accessi, consigliamo di utilizzare AWS IAM Identity Center. È possibile creare utenti e gruppi in IAM Identity Center oppure connettersi e sincronizzarsi con un gruppo di utenti e gruppi nell'origine di identità per utilizzarli in tutte le applicazioni e gli Account AWS. Per ulteriori informazioni sul Centro identità IAM, consulta Cos'è Centro identità IAM? nella Guida per l'utente di AWS IAM Identity Center.

Utenti e gruppi IAM

Un utente IAM è una identità all'interno del tuo Account AWS che dispone di autorizzazioni specifiche per una singola persona o applicazione. Ove possibile, consigliamo di fare affidamento a credenziali temporanee invece di creare utenti IAM con credenziali a lungo termine come le password e le chiavi di accesso. Tuttavia, per casi d'uso specifici che richiedono credenziali a lungo termine con utenti IAM, si consiglia di ruotare le chiavi di accesso. Per ulteriori informazioni, consulta la pagina Rotazione periodica delle chiavi di accesso per casi d'uso che richiedono credenziali a lungo termine nella Guida per l'utente di IAM.

Un gruppo IAM è un'identità che specifica un insieme di utenti IAM. Non è possibile eseguire l'accesso come gruppo. È possibile utilizzare gruppi per specificare le autorizzazioni per più utenti alla volta. I gruppi semplificano la gestione delle autorizzazioni per set di utenti di grandi dimensioni. Ad esempio, è possibile avere un gruppo denominato Amministratori IAM e concedere a tale gruppo le autorizzazioni per amministrare le risorse IAM.

Gli utenti sono diversi dai ruoli. Un utente è associato in modo univoco a una persona o un'applicazione, mentre un ruolo è destinato a essere assunto da chiunque ne abbia bisogno. Gli

utenti dispongono di credenziali a lungo termine permanenti, mentre i ruoli forniscono credenziali temporanee. Per ulteriori informazioni, consulta Quando creare un utente IAM (invece di un ruolo) nella Guida per l'utente di IAM.

Ruoli IAM

Un ruolo IAM è un'identità all'interno di un Account AWS che dispone di autorizzazioni specifiche. È simile a un utente IAM, ma non è associato a una persona specifica. È possibile assumere temporaneamente un ruolo IAM nella AWS Management Console mediante lo scambio di ruoli. È possibile assumere un ruolo chiamando un'azione AWS CLI o API AWS oppure utilizzando un URL personalizzato. Per ulteriori informazioni sui metodi per l'utilizzo dei ruoli, consulta Utilizzo di ruoli IAM nella Guida per l'utente di IAM.

I ruoli IAM con credenziali temporanee sono utili nelle seguenti situazioni:

- Accesso utente federato: per assegnare le autorizzazioni a una identità federata, è possibile creare un ruolo e definire le autorizzazioni per il ruolo. Quando un'identità federata viene autenticata, l'identità viene associata al ruolo e ottiene le autorizzazioni da esso definite. Per ulteriori informazioni sulla federazione dei ruoli, consulta Creazione di un ruolo per un provider di identità di terza parte nella Guida per l'utente di IAM. Se utilizzi IAM Identity Center, configura un set di autorizzazioni. IAM Identity Center mette in correlazione il set di autorizzazioni con un ruolo in IAM per controllare a cosa possono accedere le identità dopo l'autenticazione. Per ulteriori informazioni sui set di autorizzazioni, consulta Set di autorizzazioni nella Guida per l'utente di AWS IAM Identity Center.
- Autorizzazioni utente IAM temporanee: un utente IAM o un ruolo può assumere un ruolo IAM per ottenere temporaneamente autorizzazioni diverse per un'attività specifica.
- Accesso multi-account: è possibile utilizzare un ruolo IAM per permettere a un utente (un principale affidabile) con un account diverso di accedere alle risorse nell'account. I ruoli sono lo strumento principale per concedere l'accesso multi-account. Tuttavia, per alcuni dei Servizi AWS, è possibile collegare una policy direttamente a una risorsa (anziché utilizzare un ruolo come proxy). Per informazioni sulle differenze tra ruoli e policy basate su risorse per l'accesso multi-account, consulta Differenza tra i ruoli IAM e le policy basate su risorse nella Guida per l'utente di IAM.
- Accesso multi-servizio: alcuni Servizi AWS utilizzano funzionalità in altri Servizi AWS. Ad esempio, quando effettui una chiamata in un servizio, è comune che tale servizio esegua applicazioni in Amazon EC2 o archivi oggetti in Amazon S3. Un servizio può eseguire questa operazione utilizzando le autorizzazioni dell'entità chiamante, utilizzando un ruolo di servizio o utilizzando un ruolo collegato al servizio.

- Inoltro delle sessioni di accesso (FAS): quando si utilizza un utente o un ruolo IAM per eseguire operazioni in AWS, tale utente o ruolo viene considerato un principale. Quando si utilizzano alcuni servizi, è possibile eseguire un'operazione che attiva un'altra azione in un servizio diverso. FAS utilizza le autorizzazioni del principale che effettua la chiamata a un Servizio AWS, combinate con il Servizio AWS richiedente, per effettuare richieste a servizi a valle. Le richieste FAS vengono effettuate solo quando un servizio riceve una richiesta che necessita di interazioni con altri Servizi AWS o risorse per essere portata a termine. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le operazioni. Per i dettagli delle policy relative alle richieste FAS, consulta la pagina Forward access sessions.
- Ruolo di servizio: un ruolo di servizio è un ruolo IAM assunto da un servizio per eseguire operazioni per conto dell'utente. Un amministratore IAM può creare, modificare ed eliminare un ruolo di servizio dall'interno di IAM. Per ulteriori informazioni, consulta la sezione Creazione di un ruolo per delegare le autorizzazioni a un Servizio AWS nella Guida per l'utente di IAM.
- Ruolo collegato al servizio: un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un Servizio AWS. Il servizio può assumere il ruolo per eseguire un'azione per tuo conto. I ruoli collegati ai servizi sono visualizzati nell'account Account AWS e sono di proprietà del servizio. Un amministratore IAM può visualizzare le autorizzazioni per i ruoli collegati ai servizi, ma non modificarle.

- Applicazioni in esecuzione su Amazon EC2: è possibile utilizzare un ruolo IAM per gestire credenziali temporanee per le applicazioni in esecuzione su un'istanza EC2 che eseguono richieste di AWS CLIo dell'API AWS. Ciò è preferibile all'archiviazione delle chiavi di accesso nell'istanza EC2. Per assegnare un ruolo AWS a un'istanza EC2, affinché sia disponibile per tutte le relative applicazioni, puoi creare un profilo dell'istanza collegato all'istanza. Un profilo dell'istanza contiene il ruolo e consente ai programmi in esecuzione sull'istanza EC2 di ottenere le credenziali temporanee. Per ulteriori informazioni, consulta Utilizzo di un ruolo IAM per concedere autorizzazioni ad applicazioni in esecuzione su istanze di Amazon EC2 nella Guida per l'utente di IAM.

Per informazioni sull'utilizzo dei ruoli IAM, consulta Quando creare un ruolo IAM (invece di un utente) nella Guida per l'utente di IAM.

Gestione dell'accesso con policy

Per controllare l'accesso a AWS è possibile creare policy e collegarle a identità o risorse AWS. Una policy è un oggetto in AWS che, quando associato a un'identità o a una risorsa, ne definisce le autorizzazioni. AWS valuta queste policy quando un principale IAM (utente, utente root o sessione

ruolo) effettua una richiesta. Le autorizzazioni nelle policy determinano l'approvazione o il rifiuto della richiesta. La maggior parte delle policy viene archiviata in AWSsotto forma di documenti JSON. Per ulteriori informazioni sulla struttura e sui contenuti dei documenti delle policy JSON, consulta Panoramica delle policy JSON nella Guida per l'utente di IAM.

Gli amministratori possono utilizzare le policy AWSJSON per specificare l'accesso ai diversi elementi. In altre parole, quale principale può eseguire azioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Per concedere agli utenti l'autorizzazione a eseguire azioni sulle risorse di cui hanno bisogno, un amministratore IAM può creare policy IAM. Successivamente l'amministratore può aggiungere le policy IAM ai ruoli e gli utenti possono assumere i ruoli.

Le policy IAM definiscono le autorizzazioni relative a un'operazione, a prescindere dal metodo utilizzato per eseguirla. Ad esempio, supponiamo di disporre di una policy che consente l'azione `iam:GetRole`. Un utente con tale policy può ottenere informazioni sul ruolo dalla AWS Management Console, la AWS CLI o l'API AWS.

Policy basate su identità

Le policy basate su identità sono documenti di policy di autorizzazione JSON che è possibile allegare a un'identità (utente, gruppo di utenti o ruolo IAM). Tali policy definiscono le azioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una policy basata su identità, consulta Creazione di policy IAM nella Guida per l'utente di IAM.

Le policy basate su identità possono essere ulteriormente classificate come policy inline o policy gestite. Le policy inline sono incorporate direttamente in un singolo utente, gruppo o ruolo. Le policy gestite sono policy autonome che possono essere collegate a più utenti, gruppi e ruoli in Account AWS. Le policy gestite includono le policy gestite da AWS e le policy gestite dal cliente. Per informazioni su come scegliere tra una policy gestita o una policy inline, consulta Scelta fra policy gestite e policy inline nella Guida per l'utente di IAM.

Policy basate su risorse

Le policy basate su risorse sono documenti di policy JSON che è possibile allegare a una risorsa. Gli esempi più comuni di policy basate su risorse sono le policy di attendibilità dei ruoli IAM e le policy dei bucket Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarle per controllare l'accesso a una risorsa specifica. Quando è allegata a una risorsa, una policy definisce le azioni che un principale può eseguire su tale risorsa e a quali

condizioni. È necessario specificare un principale in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o Servizi AWS.

Le policy basate sulle risorse sono policy inline che si trovano in tale servizio. Non è possibile utilizzare le policy gestite da AWS da IAM in una policy basata su risorse.

Liste di controllo degli accessi (ACL)

Le liste di controllo degli accessi (ACL) controllano quali principali (membri, utenti o ruoli dell'account) hanno le autorizzazioni per accedere a una risorsa. Le ACL sono simili alle policy basate sulle risorse, sebbene non utilizzino il formato del documento di policy JSON.

Amazon S3, AWS WAF e Amazon VPC sono esempi di servizi che supportano le ACL. Per maggiori informazioni sulle ACL, consulta Panoramica delle liste di controllo degli accessi (ACL) nella Guida per gli sviluppatori di Amazon Simple Storage Service.

Altri tipi di policy

AWS supporta altri tipi di policy meno comuni. Questi tipi di policy possono impostare il numero massimo di autorizzazioni concesse dai tipi di policy più comuni.

- Limiti delle autorizzazioni: un limite delle autorizzazioni è una funzione avanzata nella quale si imposta il numero massimo di autorizzazioni che una policy basata su identità può concedere a un'entità IAM (utente o ruolo IAM). È possibile impostare un limite delle autorizzazioni per un'entità. Le autorizzazioni risultanti sono l'intersezione delle policy basate su identità dell'entità e i relativi limiti delle autorizzazioni. Le policy basate su risorse che specificano l'utente o il ruolo nel campo `Principal` sono condizionate dal limite delle autorizzazioni. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni sui limiti delle autorizzazioni, consulta Limiti delle autorizzazioni per le entità IAM nella Guida per l'utente di IAM.
- Policy di controllo dei servizi (SCP): le SCP sono policy JSON che specificano il numero massimo di autorizzazioni per un'organizzazione o unità organizzativa (OU) in AWS Organizations. AWS Organizationsè un servizio per il raggruppamento e la gestione centralizzata degli Account AWSmultipli di proprietà dell'azienda. Se abiliti tutte le funzionalità in un'organizzazione, puoi applicare le policy di controllo dei servizi (SCP) a uno o tutti i tuoi account. La SCP limita le autorizzazioni per le entità negli account membri, compreso ogni Utente root dell'account AWS. Per ulteriori informazioni su organizzazioni e policy SCP, consulta la pagina sulle Policy di controllo dei servizi nella Guida per l'utente di AWS Organizations.
- Policy di sessione: le policy di sessione sono policy avanzate che vengono trasmesse come parametro quando si crea in modo programmatico una sessione temporanea per un ruolo o un

utente federato. Le autorizzazioni della sessione risultante sono l'intersezione delle policy basate su identità del ruolo o dell'utente e le policy di sessione. Le autorizzazioni possono anche provenire da una policy basata su risorse. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni, consulta Policy di sessione nella Guida per l'utente di IAM.

Più tipi di policy

Quando più tipi di policy si applicano a una richiesta, le autorizzazioni risultanti sono più complicate da comprendere. Per informazioni su come AWS determina se consentire una richiesta quando sono coinvolti più tipi di policy, consulta Logica di valutazione delle policy nella Guida per l'utente di IAM.

Panoramica sulla gestione degli accessi in Amazon SQS

Ogni risorsa AWS è di proprietà di un account Account AWS e le autorizzazioni necessarie per creare o accedere a una risorsa sono regolate dalle policy di autorizzazione. Un amministratore account può collegare le autorizzazioni di policy a identità IAM (utenti, gruppi e ruoli) e alcuni servizi (ad esempio Amazon SQS) supportano anche il collegamento delle policy di autorizzazione alle risorse.

> **Note**
>
> Un amministratore account (o un utente amministratore) è un utente con privilegi di amministratore. Per ulteriori informazioni, consulta Best practice IAM nella Guida per l'utente di IAM.

Nel concedere le autorizzazioni, devi specificare quali utenti riceveranno le autorizzazioni, la risorsa per cui acquisiscono le autorizzazioni e le azioni specifiche che desideri consentire sulla risorsa.

Argomenti

- Risorse e operazioni di Amazon Simple Queue Service
- Informazioni sulla proprietà delle risorse
- Gestione dell'accesso alle risorse
- Specifica degli elementi delle policy: operazioni, effetti, risorse ed entità

Risorse e operazioni di Amazon Simple Queue Service

In Amazon SQS, l'unica risorsa è la coda. In una policy, utilizza un Amazon Resource Name (ARN) per identificare la risorsa a cui si applica la policy stessa. La seguente risorsa dispone di un ARN univoco associato:

Tipo di risorsa	Formato ARN
Queue	`arn:aws:sqs:` *`region`*`:`*`account_id`*`:`*`queue_name`*

Di seguito sono elencati alcuni esempi di formato ARN per le code:

- Un ARN per una coda denominata `my_queue` nella regione Stati Uniti orientale (Ohio), appartenente all'account AWS 123456789012:

```
arn:aws:sqs:us-east-2:123456789012:my_queue
```

- Un ARN per una coda denominata `my_queue` in ciascuna delle diverse regioni che Amazon SQS supporta:

```
arn:aws:sqs:*:123456789012:my_queue
```

- Un ARN che utilizza `*` o `?` come carattere jolly per il nome della coda. Nei seguenti esempi, l'ARN corrisponde a tutte le code con prefisso `my_prefix_`:

```
arn:aws:sqs:*:123456789012:my_prefix_*
```

Puoi ottenere il valore ARN per una coda esistente chiamando l'azione `GetQueueAttributes`. Il valore dell'attributo `QueueArn` è l'ARN della coda. Per ulteriori informazioni sugli ARN, consulta ARN IAM nella Guida per l'utente IAM.

Amazon SQS offre un set di azioni che funzionano con la risorsa della coda. Per ulteriori informazioni, consulta Autorizzazioni API Amazon SQS: riferimento a operazioni e risorse.

Informazioni sulla proprietà delle risorse

L'Account AWS possiede le risorse che vengono create nell'account, indipendentemente da chi ha creato le risorse. Nello specifico, il proprietario della risorsa è l'Account AWS dell'entità principale (ovvero l'account root, un utente IAM o un ruolo IAM) che autentica la richiesta di creazione della risorsa. Negli esempi seguenti viene illustrato il funzionamento:

- Se si utilizzano le credenziali dell'account root di Account AWS per creare una coda Amazon SQS, Account AWS è il proprietario della risorsa (in Amazon SQS, la risorsa è la coda Amazon SQS).
- Se si crea un utente nell'account Account AWS e si concedono a tale utente le autorizzazioni per creare una coda, l'utente può creare la coda. Tuttavia, tieni presente che Account AWS (a cui appartiene l'utente) è il proprietario della risorsa della coda.
- Se si crea un ruolo IAM nell'account Account AWS con le autorizzazioni per creare una coda Amazon SQS, chiunque può assumere il ruolo può creare una coda. L'account Account AWS (a cui appartiene il ruolo) è il proprietario della risorsa coda.

Gestione dell'accesso alle risorse

Una policy di autorizzazioni descrive le autorizzazioni concesse agli account. Nella sezione seguente vengono descritte le opzioni disponibili per la creazione di policy relative alle autorizzazioni.

> **Note**
>
> In questa sezione viene descritto IAM nel contesto di Amazon SQS. Non vengono fornite informazioni dettagliate sul servizio IAM. Per la documentazione di IAM completa, consulta la pagina Che cos'è IAM? nella Guida per l'utente di IAM. Per informazioni sulla sintassi delle policy IAM e le rispettive descrizioni, consulta Riferimento alle policy IAM di AWS nella Guida per l'utente di IAM.

Le policy collegate a un'identità IAM vengono definite policy basate su identità (policy IAM), mentre quelle collegate a una risorsa vengono definite policy basate su risorse.

Policy basate sull'identità (policy IAM e Amazon SQS)

Sono disponibili due modi per offrire agli utenti autorizzazioni per le code Amazon SQS: tramite il sistema di policy Amazon SQS e il sistema di policy IAM. Puoi usare uno dei due sistemi, oppure

entrambi, per collegare policy a utenti o ruoli. Nella maggior parte dei casi, puoi ottenere lo stesso risultato utilizzando uno dei due sistemi. Ad esempio, puoi eseguire le operazioni seguenti:

- Collegare una policy di autorizzazione a un utente o a un gruppo nel tuo account: per concedere a un utente le autorizzazioni per creare una coda Amazon SQS, puoi associare una policy di autorizzazioni a un utente o a un gruppo a cui appartiene l'utente.
- Collegare una policy di autorizzazione a un utente in un altro account Account AWS: per concedere autorizzazioni utente per creare una coda Amazon SQS, collega una policy di autorizzazione Amazon SQS a un utente in un altro account Account AWS.

  Le autorizzazioni per più account non sono applicabili alle operazioni seguenti:

  - `AddPermission`
  - `CancelMessageMoveTask`
  - `CreateQueue`
  - `DeleteQueue`
  - `ListMessageMoveTask`
  - `ListQueues`
  - `ListQueueTags`
  - `RemovePermission`
  - `SetQueueAttributes`
  - `StartMessageMoveTask`
  - `TagQueue`
  - `UntagQueue`
- Collegare una policy di autorizzazione a un ruolo (concedere autorizzazioni multi-account): per concedere autorizzazioni multi-account, puoi collegare una policy di autorizzazioni basata su identità a un ruolo IAM. Ad esempio, l'amministratore A dell'account Account AWS può creare un ruolo per concedere autorizzazioni tra account all'account B Account AWS (o a un servizio AWS) come segue:
  - L'amministratore dell'account A crea un ruolo IAM e attribuisce una policy di autorizzazione al ruolo che concede le autorizzazioni per le risorse nell'account A.
  - L'amministratore dell'account A attribuisce una policy di attendibilità al ruolo che identifica l'account B come il principale per tale ruolo.

- L'amministratore dell'account B delega l'autorizzazione di assumere il ruolo a qualsiasi degli utenti nell'account B. In questo modo gli utenti nell'account B possono creare o accedere a code nell'account A.

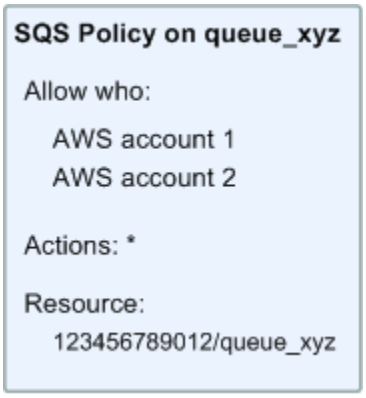
> **Note**
>
> Se desideri concedere le autorizzazioni necessarie per assumere il ruolo a un servizio AWS, il principale nella politica di attendibilità può anche essere un principale del servizio AWS.

Per ulteriori informazioni sull'uso di IAM per delegare le autorizzazioni, consulta Access Management nella IAM User Guide (Guida per l'utente di IAM).

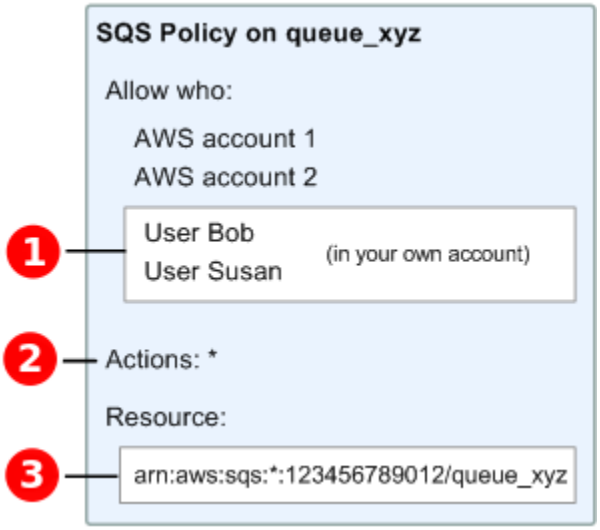
Anche se utilizza le policy IAM, Amazon SQS ha una propria infrastruttura di policy. Puoi usare una policy Amazon SQS con una coda per specificare quali account AWS possono accedere alla coda. Puoi specificare il tipo e le condizioni di accesso (per esempio, una condizione che consente di concedere autorizzazioni per l'utilizzo di `SendMessage`, `ReceiveMessage` se la richiesta viene effettuata prima del 31 dicembre 2010). Le azioni specifiche per cui puoi concedere autorizzazioni sono un sottoinsieme dell'intero elenco di azioni Amazon SQS. Quando scrivi una policy Amazon SQS e specifichi * per "consentire tutte le azioni Amazon SQS", significa che un utente può eseguire tutte le azioni in questo sottogruppo.

Il seguente diagramma mostra il concetto di una di queste policy Amazon SQS di base che copre il sottoinsieme di azioni. La policy è per `queue_xyz` e concede all'Account AWS 1 e all'Account AWS 2 le autorizzazioni per utilizzare tutte le operazioni consentite con la coda specificata.

> **Note**
>
> La risorsa nella policy è specificata come `123456789012/queue_xyz`, dove `123456789012` è l'ID account dell'account AWS proprietario della coda.



L'introduzione di IAM e dei concetti di utenti e nome della risorsa Amazon (ARN) ha prodotto alcuni cambiamenti riguardo alle policy SQS. La tabella e il diagramma seguenti descrivono le modifiche.



1

Per informazioni su come concedere le autorizzazioni agli utenti in account diversi, consulta Tutorial: Delegare l'accesso agli account AWS tramite ruoli IAM nella Guida per l'utente IAM.

2

Il sottoinsieme di operazioni incluse in * si è ampliato. Per un elenco di operazioni consentite, consulta Autorizzazioni API Amazon SQS: riferimento a operazioni e risorse.

3

Puoi specificare la risorsa utilizzando il nome della risorsa Amazon (ARN), la modalità standard di specificare le risorse in policy IAM. Per maggiori informazioni sul formato ARN per le code Amazon SQS, consulta Risorse e operazioni di Amazon Simple Queue Service.

Ad esempio, in base alla policy Amazon SQS nel diagramma precedente, chiunque possiede le credenziali di sicurezza per l'Account AWS 1 o l'Account AWS 2 può accedere a `queue_xyz`. Inoltre, gli utenti Bob e Susan nell'account AWS (con ID `123456789012`) possono accedere alla coda.

Prima dell'introduzione di IAM, Amazon SQS forniva automaticamente al creatore di una coda il controllo completo sulla stessa (ovvero, l'accesso a tutte le azioni Amazon SQS possibili su tale coda). Questo non è più vero, a meno che il creatore non usi le credenziali di sicurezza AWS. Qualsiasi utente che dispone di autorizzazioni per creare una coda deve disporre anche delle autorizzazioni per utilizzare altre azioni Amazon SQS al fine di eseguire qualsiasi operazione con le code create.

Di seguito è riportato un esempio di policy che consente a un utente di utilizzare tutte le azioni Amazon SQS, ma solo con le code i cui nomi hanno come prefisso la stringa letterale `bob_queue_`.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:*:123456789012:bob_queue_*"
   }]
}
```

Per ulteriori informazioni, consulta Utilizzo delle politiche con Amazon SQS e Identità (utenti, gruppi e ruoli) nella Guida per l'utente IAM.

Specifica degli elementi delle policy: operazioni, effetti, risorse ed entità

Per ogni risorsa Amazon Simple Queue Service, il servizio definisce un set di azioni. Per concedere le autorizzazioni per queste azioni, Amazon SQS definisce un set di azioni che puoi specificare in una policy.

> **Note**
>
> L'esecuzione di un'azione può richiedere le autorizzazioni per più di un'azione. Quando si concedono autorizzazione per operazioni specifiche, si identifica anche la risorsa per cui le operazioni sono concesse o negate.

Di seguito sono elencati gli elementi di base di una policy:

- Risorsa: in una policy si utilizza il nome della risorsa Amazon (ARN) per identificare la risorsa a cui si applica la policy stessa.
- Operazione: si utilizzano parole chiave per identificare le azioni sulla risorsa da consentire o rifiutare. Ad esempio, l'autorizzazione `sqs:CreateQueue` consente all'utente di eseguire l'azione Amazon Simple Queue Service `CreateQueue`.
- Effetto: l'effetto prodotto quando l'utente richiede l'operazione specifica, ovvero un'autorizzazione o un rifiuto. Se non concedi esplicitamente l'accesso a una risorsa, l'accesso viene implicitamente rifiutato. È anche possibile rifiutare esplicitamente l'accesso a una risorsa, per garantire che un utente non sia in grado di accedervi, anche se l'accesso viene concesso da un'altra policy.
- Principale: nelle policy basate su identità (policy IAM), l'utente a cui la policy è collegata è il principale implicito. Per policy basate su risorse, specifichi l'utente, l'account, il servizio o un'altra entità che desideri riceva le autorizzazioni (si applica solo alle policy basate su risorse).

Per ulteriori informazioni sulla sintassi e le descrizioni delle policy Amazon SQS, consulta Riferimento alle policy IAM AWS nella Guida per l'utente di IAM.

Per una tabella che mostra tutte le azioni Amazon Simple Queue Service e le risorse a cui si applicano, consulta Autorizzazioni API Amazon SQS: riferimento a operazioni e risorse.

Come funziona Amazon Simple Queue Service con IAM

Prima di utilizzare IAM per gestire l'accesso ad Amazon SQS, scopri quali funzionalità IAM sono disponibili per l'uso con Amazon SQS.

Funzionalità IAM utilizzabili con Amazon Simple Queue Service

Funzionalità IAM	Supporto per Amazon SQS
Policy basate su identità	Sì
Policy basate su risorse	Sì
Azioni di policy	Sì
Risorse relative alle policy	Sì
Chiavi di condizione della policy (specifica del servizio)	Sì

Funzionalità IAM	Supporto per Amazon SQS
Liste di controllo degli accessi (ACL)	No
ABAC (tag nelle policy)	Parziale
Credenziali temporanee	Sì
Inoltro delle sessioni di accesso (FAS)	Sì
Ruoli di servizio	Sì
Ruoli collegati al servizio	No

Per ottenere un quadro generale del funzionamento di Amazon SQS e altri servizi AWS con la maggior parte delle funzionalità di IAM, consulta Servizi AWS supportati da IAM nella Guida per l'utente IAM.

Controllo accessi

Le liste di controllo degli accessi (ACL) controllano quali principali (membri, utenti o ruoli dell'account) hanno le autorizzazioni ad accedere a una risorsa. Le ACL sono simili alle policy basate sulle risorse, sebbene non utilizzino il formato del documento di policy JSON.

Amazon S3, AWS WAF e Amazon VPC sono esempi di servizi che supportano le ACL. Per maggiori informazioni sulle ACL, consulta Panoramica delle liste di controllo degli accessi (ACL) nella Guida per gli sviluppatori di Amazon Simple Storage Service.

> **Note**
>
> È importante capire che tutti gli Account AWS possono delegare le autorizzazioni agli utenti sotto i loro account. L'accesso tra account consente di condividere l'accesso a risorse AWS senza dover gestire utenti aggiuntivi. Per ulteriori informazioni sull'utilizzo dell'accesso multi-account, consultare la sezione relativa all'abilitazione dell'accesso multiaccount nella Guida dell'utente IAM.
> Per ulteriori dettagli sulle autorizzazioni per più contenuti e sui codici di condizione all'interno delle policy personalizzate di Amazon SQS, consulta Limitazioni delle policy personalizzate.

Policy basate sull'identità per Amazon SQS

Supporta le policy basate su identità	Sì

Le policy basate su identità sono documenti di policy di autorizzazione JSON che è possibile allegare a un'identità (utente, gruppo di utenti o ruolo IAM). Tali policy definiscono le azioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una policy basata su identità, consulta Creazione di policy IAM nella Guida per l'utente di IAM.

Con le policy basate su identità di IAM, è possibile specificare quali operazioni e risorse sono consentite o respinte, nonché le condizioni in base alle quali le operazioni sono consentite o respinte. Non è possibile specificare l'entità principale in una policy basata sull'identità perché si applica all'utente o al ruolo a cui è associato. Per informazioni su tutti gli elementi utilizzabili in una policy JSON, consulta Guida di riferimento agli elementi delle policy JSON IAM nella Guida per l'utente di IAM.

Esempi di policy basate su identità per Amazon SQS

Per visualizzare degli esempi di policy basate sull'identità di Amazon SQS, consulta la pagina Best practice per le policy.

Policy basate sulle risorse all'interno di Amazon SQS

Supporta le policy basate su risorse	Sì

Le policy basate su risorse sono documenti di policy JSON che è possibile allegare a una risorsa. Gli esempi più comuni di policy basate su risorse sono le policy di attendibilità dei ruoli IAM e le policy dei bucket Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarle per controllare l'accesso a una risorsa specifica. Quando è allegata a una risorsa, una policy definisce le azioni che un principale può eseguire su tale risorsa e a quali condizioni. È necessario specificare un principale in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o Servizi AWS.

Per consentire l'accesso multi-account, puoi specificare un intero account o entità IAM in un altro account come principale in una policy basata sulle risorse. L'aggiunta di un principale multi-account

a una policy basata sulle risorse rappresenta solo una parte della relazione di trust. Quando l'entità principale e la risorsa si trovano in diversi Account AWS, un amministratore IAM nell'account attendibile deve concedere all'entità principale (utente o ruolo) anche l'autorizzazione per accedere alla risorsa. L'autorizzazione viene concessa collegando all'entità una policy basata sull'identità. Tuttavia, se una policy basata su risorse concede l'accesso a un principale nello stesso account, non sono richieste ulteriori policy basate su identità. Per ulteriori informazioni, consulta Differenza tra i ruoli IAM e le policy basate su risorse nella Guida per l'utente di IAM.

Operazioni delle policy per Amazon SQS

Supporta le azioni di policy	Sì

Gli amministratori possono utilizzare le policy JSON AWS per specificare gli accessi ai diversi elementi. Cioè, quale principale può eseguire azioni su quali risorse, e in quali condizioni.

L'elemento `Action` di una policy JSON descrive le operazioni che è possibile utilizzare per consentire o negare l'accesso a un criterio. Le azioni di policy hanno spesso lo stesso nome dell'operazione API AWS. Ci sono alcune eccezioni, ad esempio le azioni di sola autorizzazione che non hanno un'operazione API corrispondente. Esistono anche alcune operazioni che richiedono più operazioni in una policy. Queste operazioni aggiuntive sono denominate operazioni dipendenti.

Includi le operazioni in una policy per concedere le autorizzazioni a eseguire l'operazione associata.

Per un elenco di operazioni di Amazon SQS, consulta Risorse definite da Amazon Simple Queue Service nella Documentazione di riferimento all'autorizzazione del servizio.

Le azioni di policy in Amazon SQS utilizzano il seguente prefisso prima dell'azione:

```
sqs
```

Per specificare più operazioni in una sola istruzione, occorre separarle con la virgola.

```
"Action": [
      "sqs:action1",
      "sqs:action2"
         ]
```

Per alcuni esempi di policy basate sull'identità di Amazon SQS, consulta la pagina Best practice per le policy.

Risorse di policy per Amazon SQS

Supporta le risorse di policy	Sì

Gli amministratori possono utilizzare le policy JSON AWS per specificare gli accessi ai diversi elementi. Cioè, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento JSON `Resource` della policy specifica l'oggetto o gli oggetti ai quali si applica l'azione. Le istruzioni devono includere un elemento `Resource` o un elemento `NotResource`. Come best practice, specifica una risorsa utilizzando il suo nome della risorsa Amazon (ARN). Puoi eseguire questa operazione per azioni che supportano un tipo di risorsa specifico, note come autorizzazioni a livello di risorsa.

Per le azioni che non supportano le autorizzazioni a livello di risorsa, ad esempio le operazioni di elenco, utilizza un carattere jolly (*) per indicare che l'istruzione si applica a tutte le risorse.

```
"Resource": "*"
```

Per visualizzare un elenco di tipi di risorse di Amazon SQS, consulta Operazioni definite da Amazon Simple Queue Service nella documentazione di riferimento all'autorizzazione del servizio. Per informazioni sulle azioni con cui è possibile specificare l'ARN di ciascuna risorsa, consulta Risorse definite da Amazon Simple Queue Service.

Per alcuni esempi di policy basate sull'identità di Amazon SQS, consulta la pagina Best practice per le policy.

Chiavi di condizione delle policy per Amazon SQS

Supporta le chiavi di condizione delle policy specifiche del servizio	Sì

Gli amministratori possono utilizzare le policy JSON AWS per specificare gli accessi ai diversi elementi. Cioè, quale principale può eseguire azioni su quali risorse, e in quali condizioni.

L'elemento `Condition` (o blocco `Condition`) consente di specificare le condizioni in cui un'istruzione è in vigore. L'elemento `Condition` è facoltativo. Puoi compilare espressioni condizionali che utilizzano operatori di condizione, ad esempio uguale a o minore di, per soddisfare la condizione nella policy con i valori nella richiesta.

Se specifichi più elementi `Condition` in un'istruzione o più chiavi in un singolo elemento `Condition`, questi vengono valutati da AWS utilizzando un'operazione `AND` logica. Se specifichi più valori per una singola chiave di condizione, AWS valuta la condizione utilizzando un'operazione `OR` logica. Tutte le condizioni devono essere soddisfatte prima che le autorizzazioni dell'istruzione vengano concesse.

Puoi anche utilizzare variabili segnaposto quando specifichi le condizioni. Ad esempio, puoi autorizzare un utente IAM ad accedere a una risorsa solo se è stata taggata con il relativo nome utente IAM. Per ulteriori informazioni, consulta Elementi delle policy IAM: variabili e tag nella Guida per l'utente di IAM.

AWS supporta chiavi di condizione globali e chiavi di condizione specifiche per il servizio. Per visualizzare tutte le chiavi di condizione globali di AWS, consulta Chiavi di contesto delle condizioni globali di AWS nella Guida per l'utente di IAM.

Per un elenco completo delle chiavi di condizione di Amazon SQS, consulta Chiavi di condizione per Amazon Simple Queue Service nella documentazione di riferimento all'autorizzazione di servizio. Per informazioni su operazioni e risorse con cui è possibile utilizzare una chiave di condizione, consulta Risorse definite da Amazon Simple Queue Service.

Per alcuni esempi di policy basate sull'identità di Amazon SQS, consulta la pagina Best practice per le policy.

ACL in Amazon SQS

Supporta le ACL	No

Le liste di controllo degli accessi (ACL) controllano quali principali (membri, utenti o ruoli dell'account) hanno le autorizzazioni ad accedere a una risorsa. Le ACL sono simili alle policy basate su risorse, sebbene non utilizzino il formato del documento di policy JSON.

ABAC con Amazon SQS

Supporta ABAC (tag nelle policy)	Parziale

Il controllo dell'accesso basato su attributi (ABAC) è una strategia di autorizzazione che definisce le autorizzazioni in base agli attributi. In AWS, tali attributi sono denominati tag. È possibile collegare dei tag alle entità IAM (utenti o ruoli) e a numerose risorse AWS. L'assegnazione di tag alle entità e alle risorse è il primo passaggio di ABAC. In seguito, vengono progettate policy ABAC per consentire operazioni quando il tag dell'entità principale corrisponde al tag sulla risorsa a cui si sta provando ad accedere.

La strategia ABAC è utile in ambienti soggetti a una rapida crescita e aiuta in situazioni in cui la gestione delle policy diventa impegnativa.

Per controllare l'accesso basato su tag, fornisci informazioni sui tag nell'elemento condizione di una policy utilizzando le chiavi di condizione `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*o `aws:TagKeys`.

Se un servizio supporta tutte e tre le chiavi di condizione per ogni tipo di risorsa, il valore per il servizio è Yes (Sì). Se un servizio supporta tutte e tre le chiavi di condizione solo per alcuni tipi di risorsa, allora il valore sarà Parziale.

Per ulteriori informazioni su ABAC, consulta Che cos'è ABAC? nella Guida per l'utente di IAM. Per visualizzare un tutorial con i passaggi per l'impostazione di ABAC, consulta Utilizzo del controllo degli accessi basato su attributi (ABAC) nella Guida per l'utente di IAM.

Utilizzo di credenziali temporanee con Amazon SQS

Supporta le credenziali temporanee	Sì

Alcuni Servizi AWS non funzionano quando si accede utilizzando credenziali temporanee. Per ulteriori informazioni, inclusi i Servizi AWS che funzionano con le credenziali temporanee, consulta Servizi AWS supportati da IAM nella Guida per l'utente IAM.

Le credenziali temporanee sono utilizzate se si accede alla AWS Management Console utilizzando qualsiasi metodo che non sia la combinazione di nome utente e password. Ad esempio, quando accedi ad AWS utilizzando il collegamento Single Sign-On (SSO) della tua azienda, tale processo

crea in automatico credenziali temporanee. Le credenziali temporanee vengono create in automatico anche quando accedi alla console come utente e poi cambi ruolo. Per ulteriori informazioni sullo scambio dei ruoli, consulta Cambio di un ruolo (console) nella Guida per l'utente di IAM.

È possibile creare manualmente credenziali temporanee utilizzando la AWS CLIo l'API AWS. È quindi possibile utilizzare tali credenziali temporanee per accedere ad AWS. AWSconsiglia di generare le credenziali temporanee dinamicamente anziché utilizzare chiavi di accesso a lungo termine. Per ulteriori informazioni, consulta Credenziali di sicurezza provvisorie in IAM.

Sessioni di accesso diretto per Amazon SQS

Supporta sessioni di accesso diretto (FAS)	Sì

Quando si utilizza un utente o un ruolo IAM per eseguire operazioni in AWS, si viene considerati un principale. Quando si utilizzano alcuni servizi, è possibile eseguire un'azione che attiva un'altra azione in un servizio diverso. FAS utilizza le autorizzazioni del principale che effettua la chiamata a un Servizio AWS, combinate con il Servizio AWS richiedente, per effettuare richieste a servizi a valle. Le richieste FAS vengono effettuate solo quando un servizio riceve una richiesta che necessita di interazioni con altri Servizi AWS o risorse per essere portata a termine. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le operazioni. Per i dettagli delle policy relative alle richieste FAS, consulta la pagina Forward access sessions.

Ruoli di servizio per Amazon SQS

Supporta i ruoli di servizio	Sì

Un ruolo di servizio è un ruolo IAM che un servizio assume per eseguire operazioni per tuo conto. Un amministratore IAM può creare, modificare ed eliminare un ruolo di servizio dall'interno di IAM. Per ulteriori informazioni, consulta la sezione Creazione di un ruolo per delegare le autorizzazioni a un Servizio AWS nella Guida per l'utente di IAM.

> **Warning**
>
> La modifica delle autorizzazioni per un ruolo di servizio potrebbe compromettere la funzionalità di Amazon SQS. Modifica i ruoli del servizio solo quando Amazon SQS fornisce le indicazioni per farlo.

Ruoli collegati ai servizi per Amazon SQS

Supporta i ruoli collegati ai servizi	No

Un ruolo collegato ai servizi è un tipo di ruolo di servizio che è collegato a un Servizio AWS. Il servizio può assumere il ruolo per eseguire un'operazione per tuo conto. I ruoli collegati ai servizi sono visualizzati nell'account Account AWS e sono di proprietà del servizio. Un amministratore IAM può visualizzare le autorizzazioni per i ruoli collegati ai servizi, ma non modificarle.

Per ulteriori informazioni su come creare e gestire i ruoli collegati ai servizi, consulta Servizi AWS supportati da IAM. Trova un servizio nella tabella che include un `Yes` nella colonna Service-linked role (Ruolo collegato ai servizi). Scegli il collegamento Sì per visualizzare la documentazione relativa al ruolo collegato ai servizi per tale servizio.

Aggiornamenti di Amazon SQS sulle policy gestite da AWS

Per aggiungere le autorizzazioni a utenti, gruppi e ruoli, è più semplice utilizzare policy gestite da AWS piuttosto che scrivere autonomamente le policy. La creazione di policy gestite dai clienti IAM che forniscono al tuo team solo le autorizzazioni di cui ha bisogno richiede tempo e competenza. Per iniziare rapidamente, utilizza le nostre policy gestite da AWS. Queste policy coprono i casi d'uso più comuni e sono disponibili nel tuo account AWS. Per ulteriori informazioni sulle policy gestite da AWS, consulta Policy gestite da AWS nella Guida per l'utente di IAM.

I servizi AWS mantengono e aggiornano le policy gestite da AWS. Non è possibile modificare le autorizzazioni nelle policy gestite da AWS. I servizi occasionalmente aggiungono altre autorizzazioni a una policy gestita da AWS per supportare nuove funzionalità. Questo tipo di aggiornamento interessa tutte le identità (utenti, gruppi e ruoli) a cui è collegata la policy. È più probabile che i servizi aggiornino una policy gestita da AWS quando viene avviata una nuova funzionalità o quando diventano disponibili nuove operazioni. I servizi non rimuovono le autorizzazioni da una policy gestita da AWS, pertanto gli aggiornamenti delle policy non interrompono le autorizzazioni esistenti.

Inoltre, AWS supporta policy gestite per le funzioni di processi che coprono più servizi. Ad esempio, la policy ReadOnlyAccess gestita da AWS fornisce l'accesso in sola lettura a tutti i servizi e le risorse AWS. Quando un servizio avvia una nuova funzionalità, AWS aggiunge autorizzazioni di sola lettura

per nuove operazioni e risorse. Per l'elenco e la descrizione delle policy di funzione dei processi, consulta la sezione Policy gestite da AWS per funzioni di processi nella Guida per l'utente di IAM.

AWSpolitica gestita: AmazonSQS FullAccess

È possibile allegare la policy `AmazonSQSFullAccess` alle identità Amazon SQS. Questa policy concede le autorizzazioni che consentono l'accesso completo ad Amazon SQS.

Per visualizzare le autorizzazioni relative a questa politica, consulta FullAccessAmazonSQS nel Managed Policy Reference. AWS

AWSpolitica gestita: AmazonSQS ReadOnlyAccess

È possibile allegare la policy `AmazonSQSReadOnlyAccess` alle identità Amazon SQS. Questa policy concede le autorizzazioni che consentono l'accesso in sola lettura ad Amazon SQS.

Per visualizzare le autorizzazioni per questa politica, consulta ReadOnlyAccessAmazonSQS nel Managed Policy Reference. AWS

Aggiornamenti di Amazon SQS sulle policy gestite da AWS

Visualizza i dettagli sugli aggiornamenti alle policy gestite da AWS per Amazon SQS da quando questo servizio ha iniziato a tenere traccia delle modifiche. Per gli avvisi automatici sulle modifiche apportate a questa pagina, effettua la sottoscrizione al feed RSS nella pagina Cronologia dei documenti di Amazon SQS.

Modifica	Descrizione	Data
Amazon SQS ReadOnlyAccess	Amazon SQS ha aggiunto una nuova azione che consente di elencare le attività di spostamento dei messaggi più recenti (fino a 10) in una coda di origine specifica. Questa operazione è associata all'operazione API `ListMessageMoveTasks`.	9 giugno 2023

Risoluzione dei problemi di identità e accesso ad Amazon Simple Queue Service

Utilizza le informazioni seguenti per eseguire la diagnosi e risolvere i problemi comuni che possono verificarsi durante l'utilizzo di Amazon SQS e IAM.

Argomenti

Non dispongo dell'autorizzazione per eseguire un'azione in Amazon SQS

Se ricevi un errore che indica che non disponi dell'autorizzazione per eseguire un'operazione, le tue policy devono essere aggiornate in modo che ti sei consentito eseguire tale operazione.

Il seguente esempio di errore si verifica quando l'utente `mateojackson` prova a utilizzare la console per visualizzare i dettagli relativi a una risorsa *`my-example-widget`* fittizia, ma non dispone di autorizzazioni `sqs:`*`GetWidget`* fittizie.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 sqs:GetWidget on resource: my-example-widget
```

In questo caso, la policy deve essere aggiornata in modo che Mateo possa accedere alla risorsa *`my-example-widget`* mediante l'operazione `sqs:`*`GetWidget`*.

Per ulteriore assistenza con l'accesso, contatta l'amministratore AWS. L'amministratore è la persona che ti ha fornito le credenziali di accesso.

Non sono autorizzato a eseguire iam: PassRole

Se ricevi un errore che indica che non disponi dell'autorizzazione per eseguire l'azione `iam:PassRole`, per poter passare un ruolo ad Amazon SQS dovrai aggiornare le policy.

Alcuni Servizi AWSconsentono di trasmettere un ruolo esistente a tale servizio, invece di creare un nuovo ruolo di servizio o un ruolo collegato ai servizi. Per eseguire questa operazione, è necessario disporre delle autorizzazioni per trasmettere il ruolo al servizio.

Il seguente esempio di errore si verifica quando un utente IAM denominato `marymajor` cerca di utilizzare la console per eseguire un'azione in Amazon SQS. Tuttavia, l'azione richiede che il servizio disponga delle autorizzazioni concesse da un ruolo di servizio. Mary non dispone delle autorizzazioni per passare il ruolo al servizio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In questo caso, le policy di Mary devono essere aggiornate per poter eseguire l'operazione `iam:PassRole`.

Per ulteriore assistenza con l'accesso, contatta l'amministratore AWS. L'amministratore è la persona che ti ha fornito le credenziali di accesso.

Desidero consentire alle persone esterne al mio Account AWS di accedere alle mie risorse Amazon SQS

È possibile creare un ruolo con il quale utenti in altri account o persone esterne all'organizzazione possono accedere alle tue risorse. È possibile specificare chi è attendibile per l'assunzione del ruolo. Per servizi che supportano policy basate su risorse o liste di controllo accessi (ACL), utilizza tali policy per concedere alle persone l'accesso alle tue risorse.

Per ulteriori informazioni, consulta gli argomenti seguenti:

- Per sapere se Amazon SQS supporta queste funzionalità, consulta Come funziona Amazon Simple Queue Service con IAM.
- Per informazioni su come garantire l'accesso alle risorse negli Account AWSche possiedi, consulta Fornire l'accesso a un utente IAM in un altro Account AWS in tuo possesso nella Guida per l'utente IAM.
- Per informazioni su come fornire l'accesso alle risorse ad Account AWS di terze parti, consulta Fornire l'accesso agli Account AWS di proprietà di terze parti nella Guida per l'utente IAM.
- Per informazioni su come fornire l'accesso tramite la federazione delle identità, consulta Fornire l'accesso a utenti autenticati esternamente (Federazione delle identità) nella Guida per l'utente di IAM.
- Per informazioni sulle differenze tra l'utilizzo di ruoli e policy basate su risorse per l'accesso multi-account, consulta Differenza tra i ruoli IAM e le policy basate su risorse nella Guida per l'utente IAM.

Utilizzo delle politiche con Amazon SQS

In questo argomento vengono forniti esempi di policy basate su identità in cui un amministratore account può collegare policy di autorizzazione a identità IAM ovvero utenti, gruppi e ruoli.

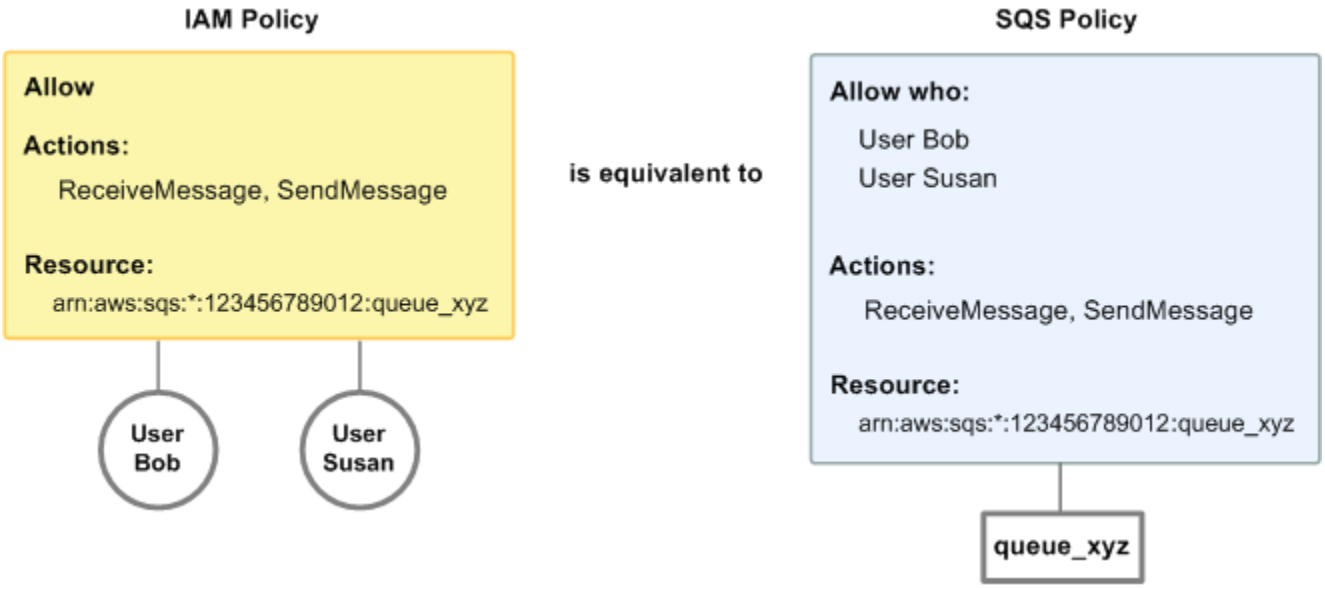
> **Important**
>
> In primo luogo, è consigliabile esaminare gli argomenti introduttivi in cui vengono spiegati i concetti di base e le opzioni disponibili per gestire l'accesso alle risorse di Amazon Simple Queue Service. Per ulteriori informazioni, consulta Panoramica sulla gestione degli accessi in Amazon SQS.
>
> Ad eccezione di `ListQueues`, tutte le azioni Amazon SQS supportano autorizzazioni a livello di risorsa. Per ulteriori informazioni, consulta Autorizzazioni API Amazon SQS: riferimento a operazioni e risorse.

Argomenti

Utilizzo di policy Amazon SQS e IAM

Sono disponibili due modi per offrire agli utenti autorizzazioni per le code Amazon SQS: utilizzando il sistema di policy Amazon SQS e il sistema di policy IAM. Puoi usare l'uno o l'altra o entrambi. Nella maggior parte dei casi, puoi ottenere gli stessi risultati con uno dei due.

Ad esempio, il diagramma seguente mostra l'equivalenza tra una policy IAM e una policy Amazon SQS. La policy IAM concede i diritti alle azioni `ReceiveMessage` e `SendMessage` di Amazon SQS per la coda chiamata `queue_xyz` nell'account AWS e la policy è collegata a utenti denominati Bob e Susan (Bob e Susan dispongono delle autorizzazioni indicate nella policy). Questa policy Amazon SQS fornisce inoltre a Bob e Susan i diritti per le azioni `ReceiveMessage` e `SendMessage` per la stessa coda.

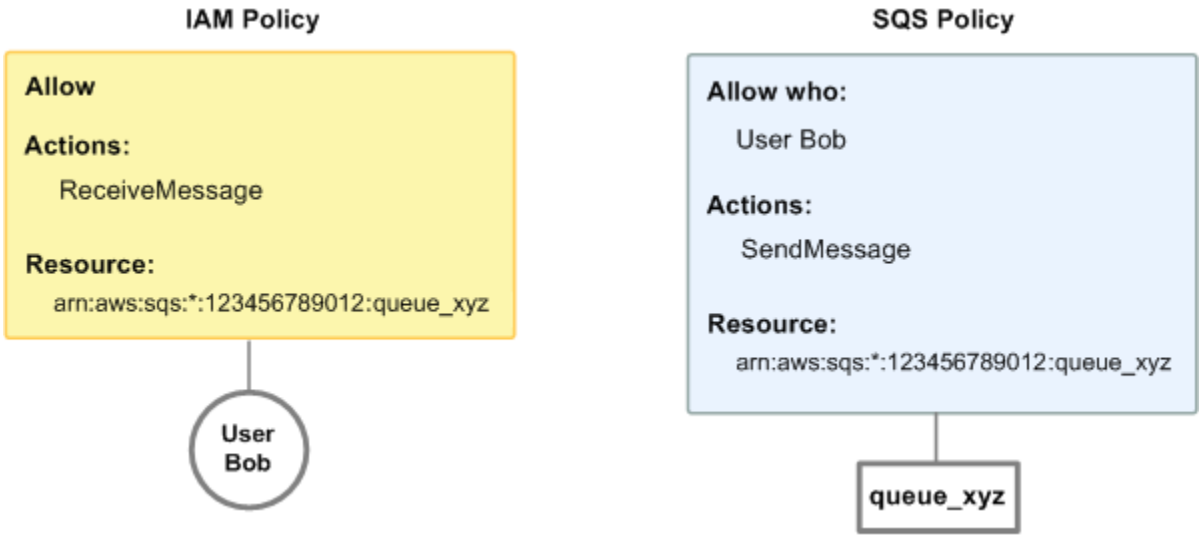


> **Note**
>
> Questo esempio mostra delle policy semplici senza condizioni. Puoi specificare una particolare condizione in una qualsiasi delle policy e ottenere lo stesso risultato.

Esiste tuttavia una grande differenza tra le policy IAM e Amazon SQS: il sistema di policy Amazon SQS, contrariamente a IAM, ti permette di concedere l'autorizzazione ad altri account AWS.

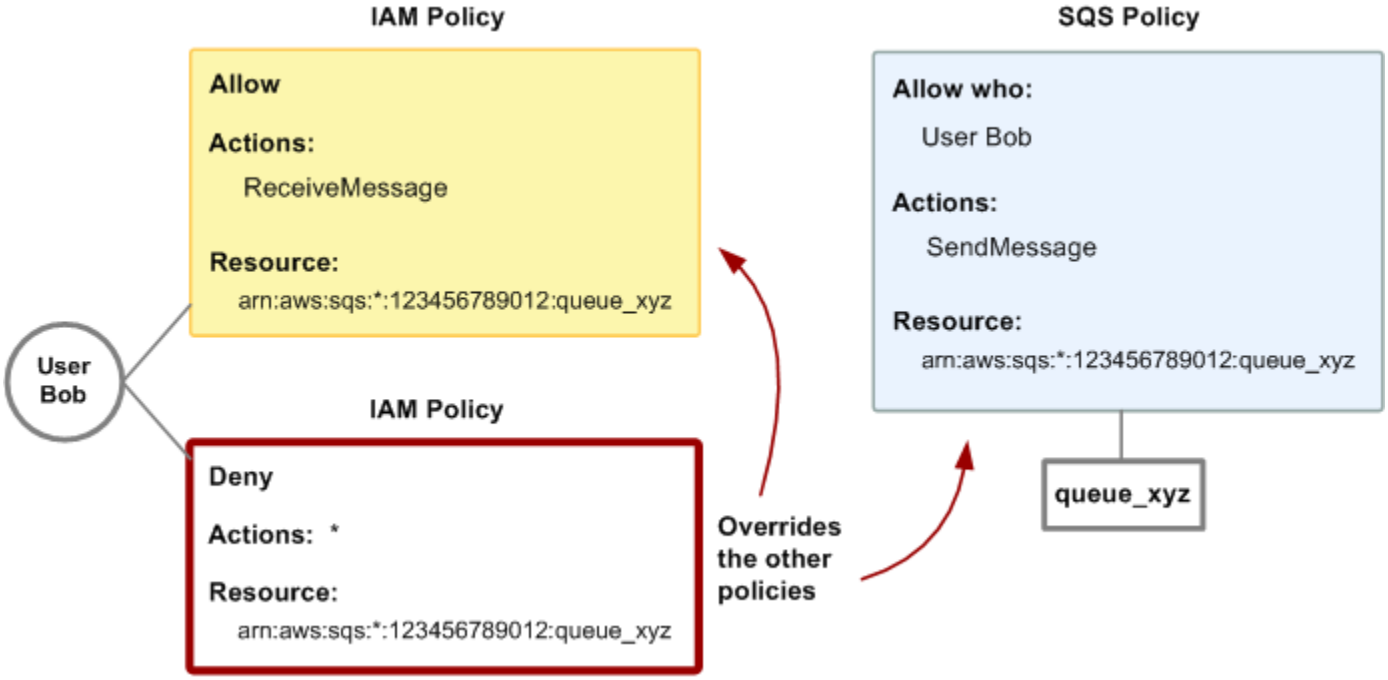
Sta a te decidere se utilizzare insieme i sistemi per gestire le autorizzazioni. Gli esempi seguenti mostrano il modo in cui i due sistemi di policy interagiscono.

- Nel primo esempio, Bob dispone sia di una policy IAM sia di una policy Amazon SQS applicabili all'account. La policy IAM concede all'account di Bob l'autorizzazione per l'azione `ReceiveMessage` su `queue_xyz`, mentre la policy Amazon SQS concede all'account l'autorizzazione per l'operazione `SendMessage` sulla stessa coda. Il diagramma seguente illustra questo concetto.



Se Bob invia una richiesta `ReceiveMessage` a `queue_xyz`, la policy IAM consente l'azione. Se Bob invia una richiesta `SendMessage` a `queue_xyz`, la policy Amazon SQS consente l'azione.

- Nel secondo esempio, Bob abusa del suo accesso a `queue_xyz`, pertanto è necessario rimuovere il suo intero accesso alla coda. La cosa più semplice da fare è aggiungere una policy che gli nega l'accesso a tutte le azioni per la coda. Questa policy sostituisce le altre due perché un `deny` esplicito sostituisce sempre un `allow`. Per ulteriori informazioni sulla logica di valutazione della policy, consulta Utilizzo di policy personalizzate con la sintassi delle policy di accesso Amazon SQS. Il diagramma seguente illustra questo concetto.



Puoi anche aggiungere un'ulteriore istruzione alla policy Amazon SQS che nega a Bob qualsiasi tipo di accesso alla coda. Ha lo stesso effetto dell'aggiunta di una policy IAM che nega a Bob l'accesso alla coda. Per esempi di policy che coprono le azioni e le risorse di Amazon SQS, consulta Esempi di base di policy Amazon SQS. Per ulteriori informazioni sulla sintassi di policy Amazon SQS, consulta Utilizzo di policy personalizzate con la sintassi delle policy di accesso Amazon SQS.

Autorizzazioni necessarie per utilizzare la console Amazon SQS

Un utente che desidera utilizzare la console Amazon SQS deve disporre di un set di autorizzazioni minimo per utilizzare le code Amazon SQS nell'account Account AWS dell'utente. Ad esempio, l'utente deve avere l'autorizzazione di chiamare l'azione `ListQueues` per essere in grado di elencare le code, o l'autorizzazione di chiamare l'azione `CreateQueue` per essere in grado di creare code. Oltre alle autorizzazioni Amazon SQS, per iscrivere una coda Amazon SQS a un argomento Amazon SNS, la console richiede anche le autorizzazioni per operazioni Amazon SNS.

Se decidi di creare una policy IAM più restrittiva relativa alle autorizzazioni minime richieste, la console potrebbe non funzionare come previsto per gli utenti con tale policy IAM.

Non sono necessarie le autorizzazioni minime della console per gli utenti che effettuano chiamate solo alle operazioni AWS CLI o Amazon SQS.

Esempi di policy basate su identità per Amazon SQS

Per impostazione predefinita, gli utenti e i ruoli IAM non dispongono dell'autorizzazione per creare o modificare risorse Amazon SQS. Inoltre, non sono in grado di eseguire attività utilizzando la AWS Management Console, l'AWS Command Line Interface (AWS CLI) o l'API AWS. Per concedere agli utenti l'autorizzazione per eseguire operazioni sulle risorse di cui hanno bisogno, un amministratore IAM può creare policy IAM. L'amministratore può quindi aggiungere le policy IAM ai ruoli e gli utenti possono assumere i ruoli.

Per informazioni su come creare una policy basata su identità IAM utilizzando questi documenti di policy JSON di esempio, consulta Creazione di policy IAM nella Guida per l'utente di IAM.

Per informazioni dettagliate sulle operazioni e sui tipi di risorse definiti da Amazon SQS, incluso il formato degli ARN per ogni tipo di risorsa, consulta Operazioni, risorse e chiavi di condizione per Amazon Simple Queue Service nella Guida di riferimento per l'autorizzazione dei servizio.

> **Note**
>
> Quando configuri gli hook del ciclo di vita per il Dimensionamento automatico Amazon EC2, non è necessario scrivere una policy per inviare messaggi a una coda Amazon SQS. Per ulteriori informazioni, consulta Hook del ciclo di vita del Dimensionamento automatico Amazon EC2 nella Guida per l'utente di Amazon EC2 per le istanze Linuxg.

Argomenti

Best practice per le policy

Le policy basate sull'identità determinano se qualcuno può creare, accedere o eliminare risorse Amazon SQS nell'account. Queste operazioni possono comportare costi aggiuntivi per l'Account AWS. Quando crei o modifichi policy basate su identità, segui queste linee guida e raccomandazioni:

- Nozioni di base sulle policy gestite da AWSe passaggio alle autorizzazioni con privilegio minimo: per le informazioni di base su come concedere autorizzazioni a utenti e carichi di lavoro, utilizza le policy gestite da AWSche concedono le autorizzazioni per molti casi d'uso comuni. Sono disponibili nel tuo Account AWS. Ti consigliamo pertanto di ridurre ulteriormente le autorizzazioni definendo policy gestite dal cliente di AWSspecifiche per i tuoi casi d'uso. Per ulteriori informazioni, consulta Policy gestite da AWS o Policy gestite da AWS per le funzioni dei processi nella Guida per l'utente IAM.
- Applica le autorizzazioni con privilegi minimi: quando imposti le autorizzazioni con le policy IAM, concedi solo le autorizzazioni richieste per eseguire un'attività. Puoi farlo definendo le azioni che possono essere intraprese su risorse specifiche in condizioni specifiche, note anche come autorizzazioni con privilegi minimi. Per ulteriori informazioni sull'utilizzo di IAM per applicare le autorizzazioni, consulta Policy e autorizzazioni in IAM nella Guida per l'utente di IAM.

- Condizioni d'uso nelle policy IAM per limitare ulteriormente l'accesso: per limitare l'accesso a operazioni e risorse puoi aggiungere una condizione alle tue policy. Ad esempio, è possibile scrivere una condizione di policy per specificare che tutte le richieste devono essere inviate utilizzando SSL. Puoi inoltre utilizzare le condizioni per concedere l'accesso alle operazioni di servizio, ma solo se vengono utilizzate tramite uno specifico Servizio AWS, ad esempio AWS CloudFormation. Per ulteriori informazioni, consulta la sezione Elementi delle policy JSON di IAM: condizione nella Guida per l'utente di IAM.
- Utilizzo di IAM Access Analyzer per convalidare le policy IAM e garantire autorizzazioni sicure e funzionali: IAM Access Analyzer convalida le policy nuove ed esistenti in modo che aderiscano alla sintassi della policy IAM (JSON) e alle best practice di IAM. IAM Access Analyzer offre oltre 100 controlli delle policy e consigli utili per creare policy sicure e funzionali. Per ulteriori informazioni, consulta Convalida delle policy per IAM Access Analyzer nella Guida per l'utente di IAM.
- Richiesta dell'autenticazione a più fattori (MFA): se hai uno scenario che richiede utenti IAM o utenti root nel tuo Account AWS, attiva MFA per una maggiore sicurezza. Per richiedere la MFA quando vengono chiamate le operazioni API, aggiungi le condizioni MFA alle policy. Per ulteriori informazioni, consulta Configurazione dell'accesso alle API protetto con MFA nella Guida per l'utente di IAM.

Per maggiori informazioni sulle best practice in IAM, consulta Best practice di sicurezza in IAM nella Guida per l'utente di IAM.

Utilizzo della console Amazon SQS

Per accedere alla console Amazon Simple Queue Service, è necessario disporre di un set di autorizzazioni minimo. Queste autorizzazioni devono consentire di elencare e visualizzare i dettagli relativi alle risorse Amazon SQS nel tuo Account AWS. Se crei una policy basata sull'identità più restrittiva rispetto alle autorizzazioni minime richieste, la console non funzionerà nel modo previsto per le entità (utenti o ruoli) associate a tale policy.

Non è necessario concedere le autorizzazioni minime della console agli utenti che effettuano chiamate solo alla AWS CLI o all'API AWS. Al contrario, concedi l'accesso solo alle operazioni che corrispondono all'operazione API che stanno cercando di eseguire.

Per garantire che utenti e ruoli possano continuare a utilizzare la console Amazon SQS, collega anche la policy gestita di Amazon `AmazonSQSReadOnlyAccess` AWS SQS alle entità. Per ulteriori informazioni, consulta Aggiunta di autorizzazioni a un utente nella Guida per l'utente IAM.

Consentire agli utenti di visualizzare le loro autorizzazioni

Questo esempio mostra in che modo è possibile creare una policy che consente agli utenti IAM di visualizzare le policy inline e gestite che sono allegate alla relativa identità utente. La policy include le autorizzazioni per completare questa azione sulla console o a livello di programmazione utilizzando la AWS CLIo l'API AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Consenti a un utente di creare code

Nell'esempio seguente creiamo una policy che consente a Bob di accedere a tutte le operazioni Amazon SQS, ma solo con code i cui nomi hanno come prefisso la stringa letterale `alice_queue_`.

Amazon SQS non concede automaticamente al creatore di una coda le autorizzazioni per utilizzare la coda. Pertanto, dobbiamo concedere esplicitamente a Bob le autorizzazioni per l'utilizzo di tutte le azioni Amazon SQS, oltre all'azione `CreateQueue` nella policy IAM.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:*:123456789012:alice_queue_*"
   }]
}
```

Consenti agli sviluppatori di scrivere messaggi in una coda condivisa

In questo esempio creiamo un gruppo per sviluppatori e colleghiamo una policy che consente al gruppo di utilizzare l'operazione `SendMessage` Amazon SQS, ma solo con la coda che appartiene all'Account AWS specificato ed è denominata `MyCompanyQueue`.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "sqs:SendMessage",
      "Resource": "arn:aws:sqs:*:123456789012:MyCompanyQueue"
   }]
}
```

È possibile utilizzare `*` invece di `SendMessage` per concedere le seguenti operazioni a un principale su una coda condivisa: `ChangeMessageVisibility`, `DeleteMessage`, `GetQueueAttributes`, `GetQueueUrl`, `ReceiveMessage` e `SendMessage`.

> **Note**
>
> Sebbene * includa l'accesso fornito da altri tipi di autorizzazione, Amazon SQS considera le autorizzazioni separatamente. Ad esempio, è possibile concedere le autorizzazioni * e `SendMessage` a un utente, anche se un * include l'accesso fornito da `SendMessage`. Questo concetto si applica anche quando rimuovi un'autorizzazione. Se un principale dispone solo di un'autorizzazione *, la richiesta di rimuovere un'autorizzazione `SendMessage`, non lascia al principale un'autorizzazione di tipo tutto tranne. Al contrario, la richiesta non ha effetto, perché il principale non dispone di un'autorizzazione `SendMessage` esplicita. Per lasciare al principale solo l'autorizzazione `ReceiveMessage`, aggiungi prima l'autorizzazione `ReceiveMessage`, quindi rimuovi l'autorizzazione *.

Consenti ai manager di ottenere la dimensione generale delle code

In questo esempio creiamo un gruppo per responsabili e colleghiamo una policy che consente al gruppo di utilizzare l'azione `GetQueueAttributes` di Amazon SQS con tutte le code che appartengono all'account AWS specificato.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "sqs:GetQueueAttributes",
      "Resource": "*"
   }]
}
```

Consenti a un partner di inviare messaggi a una coda specifica

Puoi eseguire questa operazione utilizzando una policy Amazon SQS o IAM. Se il partner dispone di un Account AWS, potrebbe risultare più facile utilizzare una policy Amazon SQS. Tuttavia, qualsiasi utente nell'azienda del partner che dispone delle credenziali di sicurezza AWS può inviare messaggi alla coda. Se desideri limitare l'accesso a un determinato utente o applicazione, devi trattare il partner come un utente della tua azienda e utilizzare una policy IAM anziché una policy Amazon SQS.

In questo esempio vengono effettuate le seguenti operazioni:

1. Crea un gruppo chiamato WidgetCo a rappresentare l'azienda partner.

2. Creazione di un utente per l'utente o l'applicazione specifica presso l'azienda del partner che ha bisogno di accedere.
3. Aggiungere l'utente al gruppo .
4. Collegamento di una policy che concedere al gruppo l'accesso solo all'azione `SendMessage` solo per la coda denominata `WidgetPartnerQueue`.

```
{
   "Version": "2012-10-17",
   "Statement": [{
         "Effect": "Allow",
         "Action": "sqs:SendMessage",
         "Resource": "arn:aws:sqs:*:123456789012:WidgetPartnerQueue"
   }]
}
```

Esempi di base di policy Amazon SQS

Questa sezione mostra le policy di esempio per i casi d'uso Amazon SQS più comuni.

Puoi utilizzare la console per verificare gli effetti di ogni policy quando la alleghi all'utente. Inizialmente l'utente non dispone di alcuna autorizzazione e non sarà in grado di eseguire alcuna operazione nella console. Quando le policy vengono collegate all'utente, è possibile verificare che l'utente possa eseguire diverse operazioni nella console.

> **Note**
>
> Si consiglia di utilizzare due finestre del browser, una per concedere le autorizzazioni, l'altra per accedere alla AWS Management Console tramite le credenziali dell'utente per verificare le autorizzazioni concesse.

Esempio 1: Concedere un'autorizzazione a un Account AWS

La policy di esempio seguente concede all'Account AWS numero `111122223333` l'autorizzazione `SendMessage` per la coda denominata `444455556666/queue1` nella regione Stati Uniti orientali (Ohio).

```
{
```

```
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_SendMessage",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333"
         ]
      },
      "Action": "sqs:SendMessage",
      "Resource": "arn:aws:sqs:us-east-2:444455556666:queue1"
   }]
}
```

Esempio 2: Concedere due autorizzazioni a un Account AWS

La policy di esempio seguente concede all'Account AWS numero `111122223333` sia l'autorizzazione `SendMessage` che `ReceiveMessage` per la coda denominata `444455556666/queue1`.

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_Send_Receive",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333"
         ]
      },
      "Action": [
         "sqs:SendMessage",
         "sqs:ReceiveMessage"
      ],
      "Resource": "arn:aws:sqs:*:444455556666:queue1"
   }]
}
```

Esempio 3: Concedere tutte le autorizzazioni a due Account AWS

La policy di esempio seguente concede a due diversi numeri Account AWS (`111122223333` e `444455556666`) l'autorizzazione per utilizzare tutte le operazioni cui Amazon SQS consente

l'accesso condiviso per la coda denominata `123456789012/queue1` nella regione Stati Uniti orientali (Ohio).

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_AllActions",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333",
            "444455556666"
         ]
      },
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:us-east-2:123456789012:queue1"
   }]
}
```

Esempio 4: Concedere autorizzazioni multi-account a un ruolo e un nome utente

La policy di esempio seguente concede a `role1` e `username1` nell'Account AWS numero `111122223333` l'autorizzazione tra account per utilizzare tutte le azioni alle quali Amazon SQS consente l'accesso condiviso per la coda denominata `123456789012/queue1` nella regione Stati Uniti orientali (Ohio).

Le autorizzazioni per più account non sono applicabili alle operazioni seguenti:

- `AddPermission`
- `CancelMessageMoveTask`
- `CreateQueue`
- `DeleteQueue`
- `ListMessageMoveTask`
- `ListQueues`
- `ListQueueTags`
- `RemovePermission`
- `SetQueueAttributes`

- StartMessageMoveTask
- TagQueue
- UntagQueue

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_AllActions",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "arn:aws:iam::111122223333:role/role1",
            "arn:aws:iam::111122223333:user/username1"
         ]
      },
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:us-east-2:123456789012:queue1"
   }]
}
```

Esempio 5: Concedere un'autorizzazione a tutti gli utenti

La seguente policy di esempio concede a tutti gli utenti (utenti anonimi) l'autorizzazione `ReceiveMessage` per la coda denominata `111122223333/queue1`.

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_AnonymousAccess_ReceiveMessage",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "sqs:ReceiveMessage",
      "Resource": "arn:aws:sqs:*:111122223333:queue1"
   }]
}
```

Esempio 6: Concedere un'autorizzazione limitata nel tempo a tutti gli utenti

La seguente policy di esempio concede a tutti gli utenti (utenti anonimi) l'autorizzazione `ReceiveMessage` per la coda denominata `111122223333/queue1`, ma solo tra le ore 12 (mezzogiorno) e le ore 15 del 31 gennaio 2009.

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_AnonymousAccess_ReceiveMessage_TimeLimit",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "sqs:ReceiveMessage",
      "Resource": "arn:aws:sqs:*:111122223333:queue1",
      "Condition" : {
         "DateGreaterThan" : {
            "aws:CurrentTime":"2009-01-31T12:00Z"
         },
         "DateLessThan" : {
            "aws:CurrentTime":"2009-01-31T15:00Z"
         }
      }
   }]
}
```

Esempio 7: Concedere tutte le autorizzazioni a tutti gli utenti in un intervallo CIDR

La seguente policy di esempio concede a tutti gli utenti (anonimi) l'autorizzazione a utilizzare tutte le azioni Amazon SQS possibili che possono essere condivise per la coda denominata `111122223333/queue1`, ma solo se la richiesta proviene dall'intervallo CIDR `192.0.2.0/24`.

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_AnonymousAccess_AllActions_AllowlistIP",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:*:111122223333:queue1",
      "Condition" : {
         "IpAddress" : {
```

```
            "aws:SourceIp":"192.0.2.0/24"
         }
      }
   }]
}
```

Esempio 8: Autorizzazioni per l'elenco di elementi consentiti e l'elenco di elementi bloccati per utenti in diversi intervalli CIDR

La policy di esempio seguente dispone di due istruzioni:

- La prima dichiarazione concede a tutti gli utenti (utenti anonimi) nell'intervallo CIDR `192.0.2.0/24` (fatta eccezione per `192.0.2.188`) l'autorizzazione per utilizzare l'azione `SendMessage` per la coda denominata `111122223333`/queue1.
- La seconda dichiarazione blocca tutti gli utenti (utenti anonimi) nell'intervallo CIDR `12.148.72.0/23` in modo che non possano utilizzare la coda.

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_AnonymousAccess_SendMessage_IPLimit",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "sqs:SendMessage",
      "Resource": "arn:aws:sqs:*:111122223333:queue1",
      "Condition" : {
         "IpAddress" : {
            "aws:SourceIp":"192.0.2.0/24"
         },
         "NotIpAddress" : {
            "aws:SourceIp":"192.0.2.188/32"
         }
      }
   }, {
      "Sid":"Queue1_AnonymousAccess_AllActions_IPLimit_Deny",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:*:111122223333:queue1",
      "Condition" : {
```

```
         "IpAddress" : {
            "aws:SourceIp":"12.148.72.0/23"
         }
      }
   }]
}
```

Utilizzo di policy personalizzate con la sintassi delle policy di accesso Amazon SQS

Se desideri consentire l'accesso ad Amazon SQS basato solo su un ID account Account AWS e autorizzazioni di base (ad esempio per `SendMessage` o `ReceiveMessage`), non occorre scrivere le policy. Puoi semplicemente utilizzare l'azione Amazon SQS `AddPermission`.

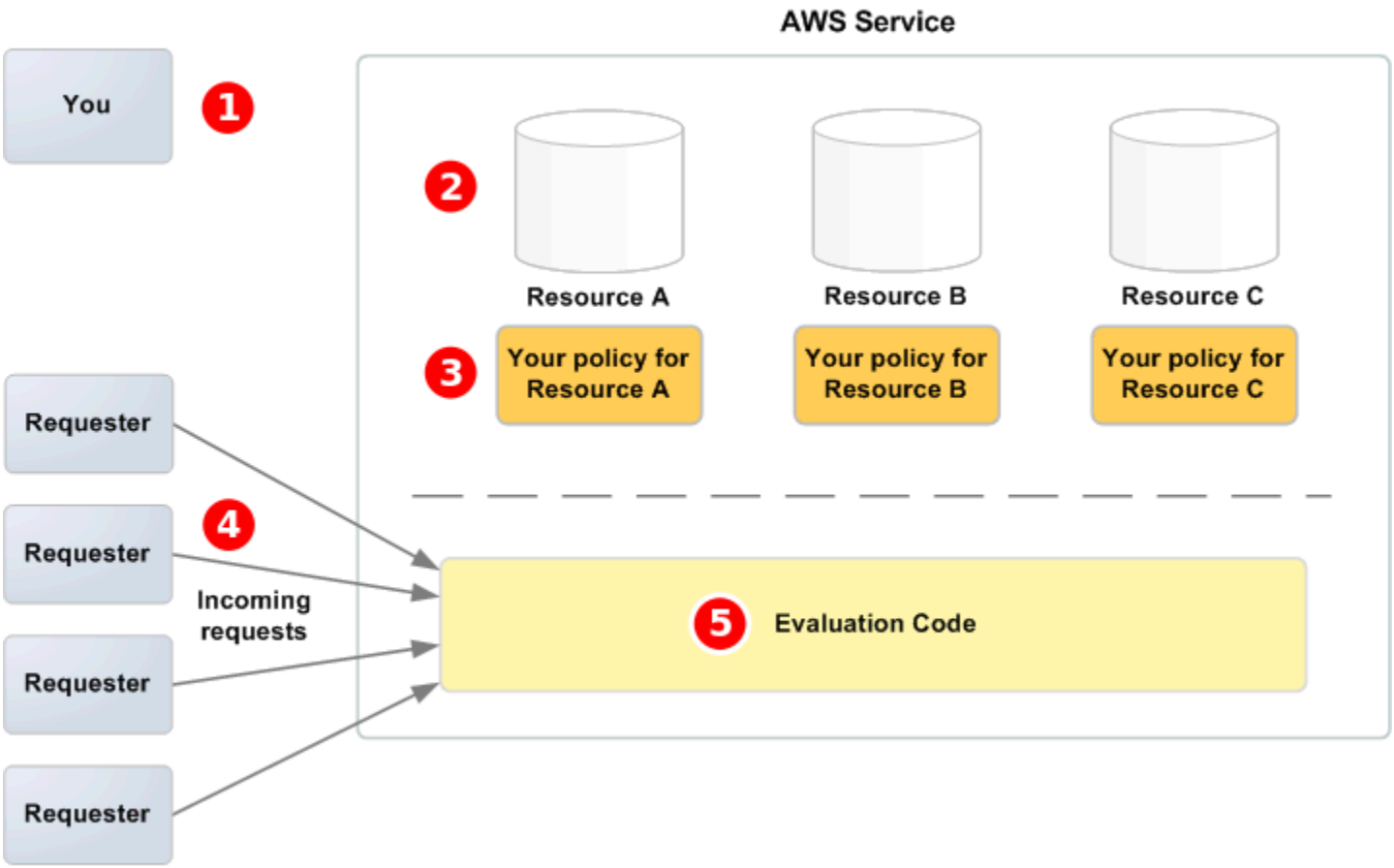
Se desideri negare o consentire esplicitamente l'accesso in base a più condizioni specifiche (ad esempio l'ora di arrivo di una richiesta o l'indirizzo IP del richiedente), devi scrivere le tue policy Amazon SQS e caricarle nel sistema AWS utilizzando l'azione Amazon SQS `SetQueueAttributes`.

Argomenti

- Architettura di controllo degli accessi Amazon SQS
- Flusso di lavoro del processo di controllo degli accessi Amazon SQS
- Concetti chiave della sintassi delle policy di accesso di Amazon SQS
- Logica di valutazione della sintassi delle policy di accesso Amazon SQS
- Relazioni tra negazioni esplicite e predefinite nella sintassi delle policy di accesso di Amazon SQS
- Limitazioni delle policy personalizzate
- Esempi di sintassi delle policy di accesso di Amazon SQS personalizzata

Architettura di controllo degli accessi Amazon SQS

Il diagramma riportato di seguito descrive il controllo degli accessi per le risorse Amazon SQS.



1

Tu, il proprietario delle risorse.

2

Le tue risorse contenute nel servizio AWS (ad esempio le code Amazon SQS).

3

Le tue policy. È buona norma avere una policy per ogni risorsa. Il servizio AWS stesso fornisce un'API che utilizzi per caricare e gestire le tue policy.

4

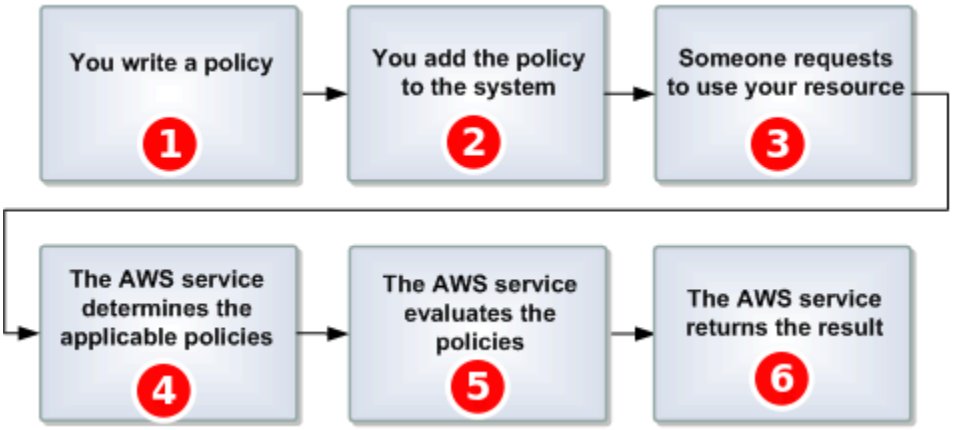
I richiedenti e le relative richieste in arrivo al servizio AWS.

5

Codice di valutazione della sintassi delle policy di accesso. Si tratta del set di codici interni al servizio AWS che valuta le richieste in arrivo rispetto alle policy applicabili e determina se il richiedente può accedere alla risorsa.

Flusso di lavoro del processo di controllo degli accessi Amazon SQS

Nel seguente diagramma viene descritto il flusso di lavoro di controllo degli accessi generale con la sintassi delle policy di accesso Amazon SQS.



1

Scrivi una policy Amazon SQS per la tua coda.

2

Carica la policy in AWS. Il servizio AWS fornisce un'API che puoi utilizzare per caricare le policy. Ad esempio, usa l'azione `SetQueueAttributes` di Amazon SQS per caricare una policy per una particolare coda Amazon SQS.

3

Qualcuno invia una richiesta per usare la tua coda Amazon SQS.

4

Amazon SQS esamina tutte le policy Amazon SQS disponibili e determina quali sono applicabili.

5

Amazon SQS valuta le policy e determina se il richiedente è autorizzato a utilizzare la tua coda o meno.

6

In base al risultato della valutazione della policy, Amazon SQS restituisce un errore `Access denied` al richiedente o continua a elaborare la richiesta.

Concetti chiave della sintassi delle policy di accesso di Amazon SQS

Per scrivere le tue proprie policy, devi avere familiarità con JSON e una serie di concetti fondamentali.

Abilita

Il risultato di una Statement che Effetto impostato su `allow`.

Action

L'attività che Principale ha l'autorizzazione di eseguire, in genere una richiesta a AWS.

Default-deny

Il risultato di una Statement priva di impostazioni Abilita o Explicit-deny.

Condition

Qualsiasi restrizione o dettaglio relativi a una Autorizzazione. Condizioni tipiche relative a data e ora e indirizzi IP.

Effetto

Il risultato che desideri che la Statement di una Policy restituisca al momento della valutazione. Puoi specificare il valore `deny` o `allow` quando scrivi la dichiarazione della policy. Possono esserci tre risultati possibili al momento della valutazione della policy: Default-deny, Abilita e Explicit-deny.

Explicit-deny

Il risultato di una Statement che Effetto impostato su `deny`.

Valutazione

Il processo che Amazon SQS utilizza per determinare se una richiesta in entrata deve essere negata o consentita in base a una Policy.

Approvatore

L'utente che scrive una Policy per concedere autorizzazioni a una risorsa. L'emittente, per definizione, è sempre il proprietario della risorsa. AWS non consente agli utenti di Amazon SQS di creare policy per risorse che non sono di loro proprietà.

Chiave

La caratteristica specifica che costituisce la base per la limitazione dell'accesso.

Autorizzazione

La nozione di consentire o rifiutare l'accesso a una risorsa utilizzando una Condition e una Chiave.

Policy

Il documento che agisce da container per una o più istruzioni.

Amzon SQS utilizza la policy per determinare se concedere o meno a un utente l'accesso a una risorsa.

Principale

L'utente che riceve la Autorizzazione nella Policy.

Resource (Risorsa)

L'oggetto a cui le richiesta del Principale accedono.

Statement

La descrizione formale di una singola autorizzazione, scritta nella sintassi delle policy di accesso come parte di un documento di Policy più ampio.

Richiedente

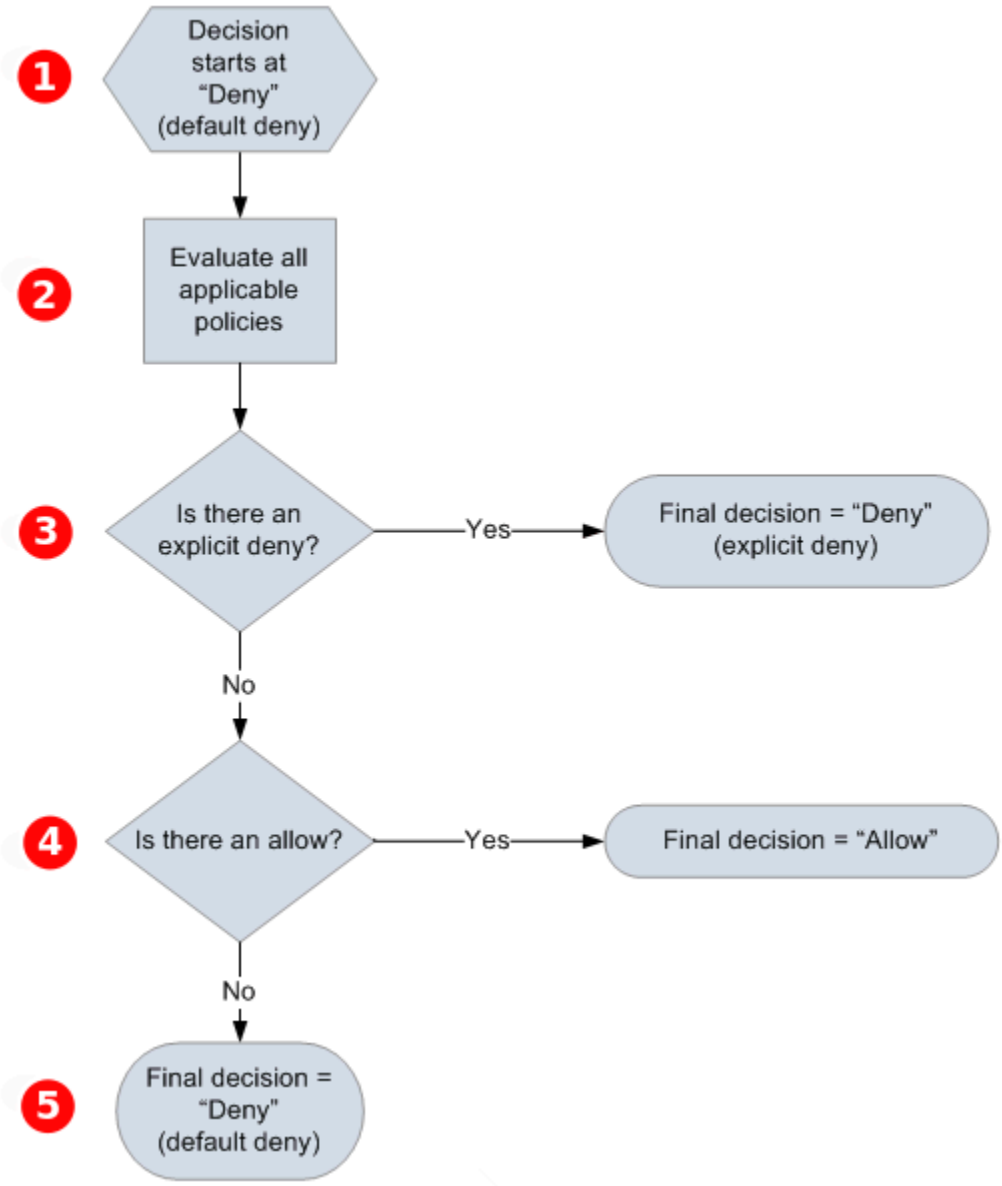
L'utente che invia una richiesta per l'accesso a una Resource (Risorsa).

Logica di valutazione della sintassi delle policy di accesso Amazon SQS

Al momento della valutazione, Amazon SQS determina se le richieste da parte di utenti diversi dal proprietario della risorsa devono essere concesse o negate. La logica della valutazione segue diverse regole di base:

- Per impostazione predefinita, tutte le richieste per utilizzare la tua risorsa che provengono da chiunque eccetto te, vengono rifiutate.
- Un Abilita sostituisce qualsiasi Default-deny.
- Un Explicit-deny sovrascrive qualsiasi allow.
- L'ordine in cui vengono valutate le policy non è importante.

Il diagramma seguente descrive in dettaglio come Amazon SQS valuta le decisioni circa le autorizzazioni di accesso.



1. La decisione inizia con un default-deny.

2. Il codice di applicazione valuta tutte le policy applicabili alla richiesta (in base alla risorsa, all'entità principale, all'operazione e alle condizioni). L'ordine in cui il codice di attuazione valuta le policy non è importante.

3. Il codice di attuazione cerca una dichiarazione explicit-deny che può essere applicata alla richiesta. Se ne trova uno, il codice di attuazione restituisce una decisione di rifiuto e il processo termina.

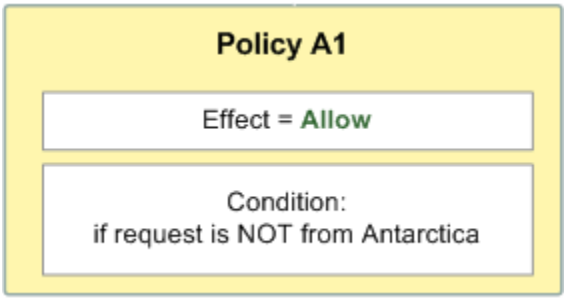
4. Se non viene trovato un explicit-deny, il codice di applicazione cerca un'istruzione allow da applicare alla richiesta. Se ne trova anche uno, il codice di attuazione restituisce una decisione di consenso e il processo termina (il servizio continua ad elaborare la richiesta).

5. Se non viene trovato alcun allow, la decisione finale è il rifiuto (dal momento che non è stato trovato un explicit-deny o un allow, questo viene considerato un default-deny).

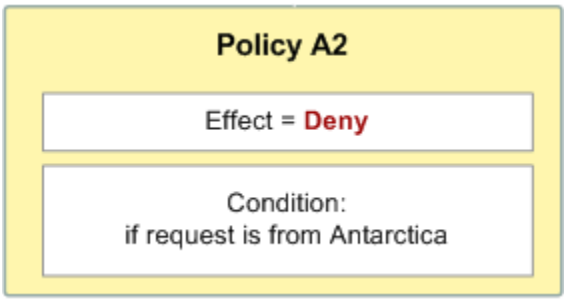
Relazioni tra negazioni esplicite e predefinite nella sintassi delle policy di accesso di Amazon SQS

Se una policy Amazon SQS non si applica direttamente a una richiesta, questa comporta un Default-deny. Ad esempio, se un utente richiede l'autorizzazione per utilizzare Amazon SQS, ma l'unica policy che si applica all'utente può utilizzare DynamoDB, la richiesta comporta un default-deny.

Se una condizione in una dichiarazione non viene soddisfatta, la richiesta comporta un default-deny. Se tutte le condizioni nella dichiarazione sono soddisfatte, la richiesta restituisce un Abilita o un Explicit-deny a seconda del valore dell'elemento Effetto della policy. Le policy non specificano cosa fare se una condizione non viene soddisfatta e quindi il risultato predefinito in quel caso è un default-deny. Ad esempio, supponiamo che tu voglia impedire le richieste provenienti dall'Antartide. Scrivi una Policy A1 che consente una richiesta solo se non proviene dall'Antartide. Il diagramma seguente illustra la policy Amazon SQS.

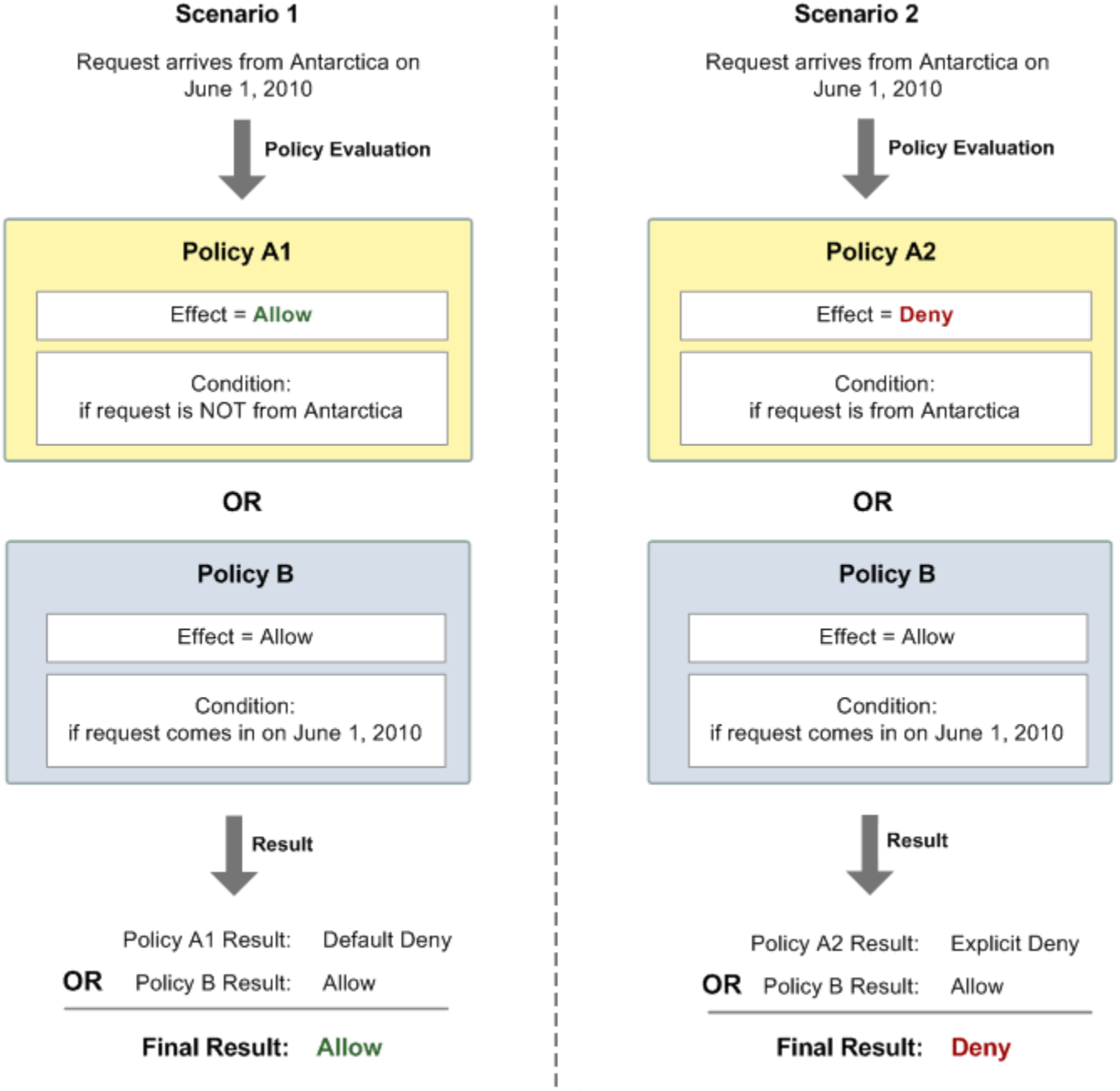


Se un utente invia una richiesta dagli Stati Uniti, la condizione è soddisfatta (la richiesta non proviene dall'Antartide) e la richiesta comporta un allow. Tuttavia, se un utente invia una richiesta dall'Antartide, la condizione non viene soddisfatta e la richiesta comporta, per impostazione predefinita, un default-deny. Puoi modificare il risultato a un explicit-deny scrivendo una Policy A2 che nega esplicitamente una richiesta se proviene da Antartide. Il diagramma seguente illustra la policy.



Se un utente invia una richiesta dall'Antartide, la condizione viene soddisfatta e la richiesta comporta un explicit-deny.

La differenza tra un default-deny e un explicit-deny è importante perché un allow può sovrascrivere il primo ma non l'ultimo. Ad esempio, Policy B consente le richieste se arrivano il 1° giugno 2010. Il seguente diagramma confronta la combinazione di questa policy con Policy A1 e Policy A2.



Nello Scenario 1, Policy A1 comporta un default-deny e Policy B comporta un allow perché la policy consente le richieste pervenute il 1° giugno 2010. Il allow dalla policy B sovrascrive il default-deny dalla policy A1 e pertanto la richiesta è consentita.

Nello Scenario 2, Policy B2 comporta un explicit-deny e Policy B comporta un allow. L'explicit-deny dalla policy A2 sovrascrive il allow dalla policy B e pertanto la richiesta viene rifiutata.

Limitazioni delle policy personalizzate

Accesso multi-account

Le autorizzazioni per più account non sono applicabili alle azioni seguenti:

- `AddPermission`
- `CancelMessageMoveTask`
- `CreateQueue`
- `DeleteQueue`
- `ListMessageMoveTask`
- `ListQueues`
- `ListQueueTags`
- `RemovePermission`
- `SetQueueAttributes`
- `StartMessageMoveTask`
- `TagQueue`
- `UntagQueue`

Chiavi di condizione

Al momento Amazon SQS supporta solo un sottoinsieme limitato delle chiavi di condizione disponibili in IAM. Per ulteriori informazioni, consulta Autorizzazioni API Amazon SQS: riferimento a operazioni e risorse.

Esempi di sintassi delle policy di accesso di Amazon SQS personalizzata

I seguenti sono esempi tipici di policy di accesso Amazon SQS.

Esempio 1: Concedere l'autorizzazione a un account

La policy Amazon SQS di esempio seguente fornisce all'account Account AWS l'autorizzazione 111122223333 per inviare e ricevere dalla `queue2` di proprietà dell'Account AWS 444455556666.

```
{
   "Version": "2012-10-17",
```

```
   "Id": "UseCase1",
   "Statement" : [{
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333"
         ]
      },
      "Action": [
         "sqs:SendMessage",
         "sqs:ReceiveMessage"
      ],
      "Resource": "arn:aws:sqs:us-east-2:444455556666:queue2"
   }]
}
```

Esempio 2: Concedere l'autorizzazione a uno o più account

La policy di esempio Amazon SQS seguente fornisce a uno o più Account AWS l'accesso a code di proprietà dell'account per un determinato periodo di tempo. È necessario scrivere questa policy e caricarla in Amazon SQS utilizzando l'operazione `SetQueueAttributes` perché l'operazione `AddPermission` non consente di specificare una limitazione di tempo quando si concede l'accesso a una coda.

```
{
   "Version": "2012-10-17",
   "Id": "UseCase2",
   "Statement" : [{
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333",
            "444455556666"
         ]
      },
      "Action": [
         "sqs:SendMessage",
         "sqs:ReceiveMessage"
      ],
      "Resource": "arn:aws:sqs:us-east-2:444455556666:queue2",
      "Condition": {
```

```
         "DateLessThan": {
            "AWS:CurrentTime": "2009-06-30T12:00Z"
         }
      }
   }]
}
```

Esempio 3: Concedere autorizzazioni a richieste provenienti da istanze Amazon EC2

La policy di esempio Amazon SQS seguente fornisce l'accesso a richieste provenienti da istanze Amazon EC2. Questo esempio si basa sull'esempio "Esempio 2: Concedere l'autorizzazione a uno o più account": limita l'accesso a prima del 30 giugno 2009 alle 12.00 (UTC), limita l'accesso all'intervallo IP `203.0.113.0/24`. È necessario scrivere questa policy e caricarla su Amazon SQS utilizzando l'azione `SetQueueAttributes` perché l'azione `AddPermission` non consente di specificare una limitazione di indirizzo IP al momento di concedere l'accesso a una coda.

```
{
   "Version": "2012-10-17",
   "Id": "UseCase3",
   "Statement" : [{
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333"
         ]
      },
      "Action": [
         "sqs:SendMessage",
         "sqs:ReceiveMessage"
      ],
      "Resource": "arn:aws:sqs:us-east-2:444455556666:queue2",
      "Condition": {
         "DateLessThan": {
            "AWS:CurrentTime": "2009-06-30T12:00Z"
         },
         "IpAddress": {
            "AWS:SourceIp": "203.0.113.0/24"
         }
      }
   }]
}
```

Esempio 4: Negare l'accesso a un account specifico

La policy Amazon SQS di esempio seguente nega a un Account AWS specifico l'accesso alla coda. Questo esempio si basa sull'esempio "Esempio 1: Concedere l'autorizzazione a un account": nega l'accesso a un Account AWS specificato. È necessario scrivere questa policy e caricarla in Amazon SQS utilizzando l'operazione `SetQueueAttributes` perché l'operazione `AddPermission` non consente di negare l'accesso a una coda (concede solo l'accesso a una coda).

```
{
   "Version": "2012-10-17",
   "Id": "UseCase4",
   "Statement" : [{
      "Sid": "1",
      "Effect": "Deny",
      "Principal": {
         "AWS": [
            "111122223333"
         ]
      },
      "Action": [
         "sqs:SendMessage",
         "sqs:ReceiveMessage"
      ],
      "Resource": "arn:aws:sqs:us-east-2:444455556666:queue2"
   }]
}
```

Esempio 5: Negare l'accesso se non è un endpoint VPC

La seguente policy Amazon SQS di esempio limita l'accesso alla `queue1`: 111122223333 può eseguire le azioni `SendMessage` e `ReceiveMessage` solo dall'ID dell'endpoint VPC `vpce-1a2b3c4d` (specificato usando la condizione `aws:sourceVpce`). Per ulteriori informazioni, consulta Endpoint di Amazon Virtual Private Cloud per Amazon SQS.

> **Note**
>
> - La condizione `aws:sourceVpce` non richiede un ARN per la risorsa dell'endpoint VPC, ma solo l'ID dell'endpoint VPC.
> - Puoi modificare l'esempio che segue per limitare tutte le operazioni per un determinato endpoint VPC negando tutte le operazioni Amazon SQS (`sqs:*`) nella seconda istruzione.

Tuttavia, questa istruzione della policy stabilisce che tutte le operazioni (comprese le operazioni amministrative necessarie per modificare le autorizzazioni della coda) devono essere eseguite tramite l'endpoint VPC specifico definito nella policy, impedendo potenzialmente all'utente di modificare successivamente le autorizzazioni della coda.

```
{
   "Version": "2012-10-17",
   "Id": "UseCase5",
   "Statement": [{
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333"
         ]
      },
      "Action": [
         "sqs:SendMessage",
         "sqs:ReceiveMessage"
      ],
         "Resource": "arn:aws:sqs:us-east-2:111122223333:queue1"
      },
      {
         "Sid": "2",
         "Effect": "Deny",
         "Principal": "*",
         "Action": [
            "sqs:SendMessage",
            "sqs:ReceiveMessage"
         ],
         "Resource": "arn:aws:sqs:us-east-2:111122223333:queue1",
         "Condition": {
            "StringNotEquals": {
               "aws:sourceVpce": "vpce-1a2b3c4d"
            }
         }
      }
   ]
}
```

Utilizzo delle credenziali di sicurezza temporanee con Amazon SQS

Oltre a creare utenti con proprie credenziali di sicurezza, IAM ti permette di assegnare credenziali di sicurezza temporanee a qualsiasi utente, autorizzandone quindi l'accesso ai tuoi servizi e risorse AWS. Puoi gestire utenti che hanno Account AWS. Puoi anche gestire utenti per il tuo sistema che non dispongono di Account AWS (utenti federati). Inoltre, gli "utenti" possono anche essere applicazioni che hai creato per accedere alle risorse AWS.

Queste credenziali di sicurezza temporanee sono utilizzate per effettuare delle richieste ad Amazon SQS. Le librerie API calcolano il valore di firma necessario utilizzando tali credenziali per autenticare la richiesta dell'utente. In caso di invio di richieste tramite l'utilizzo di credenziali scadute, Amazon SQS rifiuta la richiesta.

> **Note**
>
> Non puoi impostare una policy in base alle credenziali temporanee.

Prerequisiti

1. Usare IAM per creare credenziali di sicurezza provvisorie:
   - Token di sicurezza
   - ID chiave di accesso
   - Secret Access Key
2. Prepara la tua stringa a effettuare l'accesso con l'ID chiave di accesso temporanea e il token di sicurezza.
3. Utilizza la Secret Access Key temporanea anziché la tua Secret Access Key per firmare la tua richiesta di API query.

> **Note**
>
> Quando invii la richiesta di API query firmata, utilizza l'ID chiave di accesso temporaneo anziché l'ID chiave di accesso e includi i token di sicurezza. Per ulteriori informazioni sul supporto di IAM per le credenziali di sicurezza temporanee, consulta la pagina relativa alla Concessione dell'accesso temporaneo alle risorse AWS nella Guida per l'utente IAM.

Chiamare un'azione API query Amazon SQS utilizzando le credenziali di sicurezza temporanee

1. Richiedere un token di sicurezza provvisorio utilizzando AWS Identity and Access Management. Per ulteriori informazioni, consulta Creazione di credenziali di sicurezza temporanee per abilitare l'accesso per utenti IAM nella Guida per l'utente IAM.

   IAM restituisce un token di sicurezza, un ID chiave di accesso e una chiave di accesso segreta.
2. Prepare la tua query utilizzando l'ID chiave di accesso temporaneo anziché l'ID chiave di accesso e includi i token di sicurezza. Firma la tua richiesta utilizzando la Secret Access Key temporaneo anziché quella effettiva.
3. Invia la stringa della query firmata con l'ID chiave di accesso temporanea e il token di sicurezza.

   Nell'esempio seguente viene mostrato come utilizzare credenziali di sicurezza temporanee per autenticare una richiesta Amazon SQS. La struttura di *AUTHPARAMS* dipende dalla modalità di firma della richiesta API. Per ulteriori informazioni, consulta Firma delle richieste API AWS in Riferimento generale per Amazon Web Services.

```
https://sqs.us-east-2.amazonaws.com/
?Action=CreateQueue
&DefaultVisibilityTimeout=40
&QueueName=MyQueue
&Attribute.1.Name=VisibilityTimeout
&Attribute.1.Value=40
&Expires=2020-12-18T22%3A52%3A43PST
&SecurityToken=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE
&Version=2012-11-05
&AUTHPARAMS
```

   L'esempio seguente utilizza credenziali di sicurezza temporanee per inviare due messaggi con l'operazione `SendMessageBatch`.

```
https://sqs.us-east-2.amazonaws.com/
?Action=SendMessageBatch
&SendMessageBatchRequestEntry.1.Id=test_msg_001
&SendMessageBatchRequestEntry.1.MessageBody=test%20message%20body%201
&SendMessageBatchRequestEntry.2.Id=test_msg_002
&SendMessageBatchRequestEntry.2.MessageBody=test%20message%20body%202
&SendMessageBatchRequestEntry.2.DelaySeconds=60
&Expires=2020-12-18T22%3A52%3A43PST
```

```
&SecurityToken=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
&AWSAccessKeyId=AKIAI44QH8DHBEXAMPLE
&Version=2012-11-05
&AUTHPARAMS
```

Gestione dell'accesso alla coda crittografata di Amazon SQS utilizzando la policy di Amazon SQS con privilegi minimi e la policy della chiave AWS KMS

Puoi utilizzare Amazon SQS per lo scambio di dati sensibili tra applicazioni utilizzando la crittografia lato server (SSE) integrata con AWS Key Management Service (KMS). Con l'integrazione di Amazon SQS e AWS KMS, puoi gestire centralmente le chiavi che proteggono Amazon SQS, nonché le chiavi che proteggono le altre risorse AWS.

Diversi servizi AWS si comportano da origini eventi che possono inviare eventi alle code Amazon SQS. Per consentire a un'origine di eventi di accedere alla coda crittografata di Amazon SQS, devi configurare la coda con una chiave gestita dal clienteAWS KMS. Quindi, utilizza la policy della chiave per consentire al servizio di utilizzare i metodi API AWS KMS richiesti. Il servizio richiede inoltre le autorizzazioni per autenticare l'accesso per consentire alla coda di inviare eventi. Puoi raggiungere questo obiettivo utilizzando una policy di Amazon SQS, che è una policy basata sulle risorse che puoi utilizzare per controllare l'accesso alla coda Amazon SQS e ai relativi dati.

Le seguenti sezioni forniscono informazioni su come controllare l'accesso alla coda crittografata di Amazon SQS tramite la policy di Amazon SQS e la policy della chiave AWS KMS. Le policy di questa guida ti aiuteranno a ottenere il privilegio minimo.

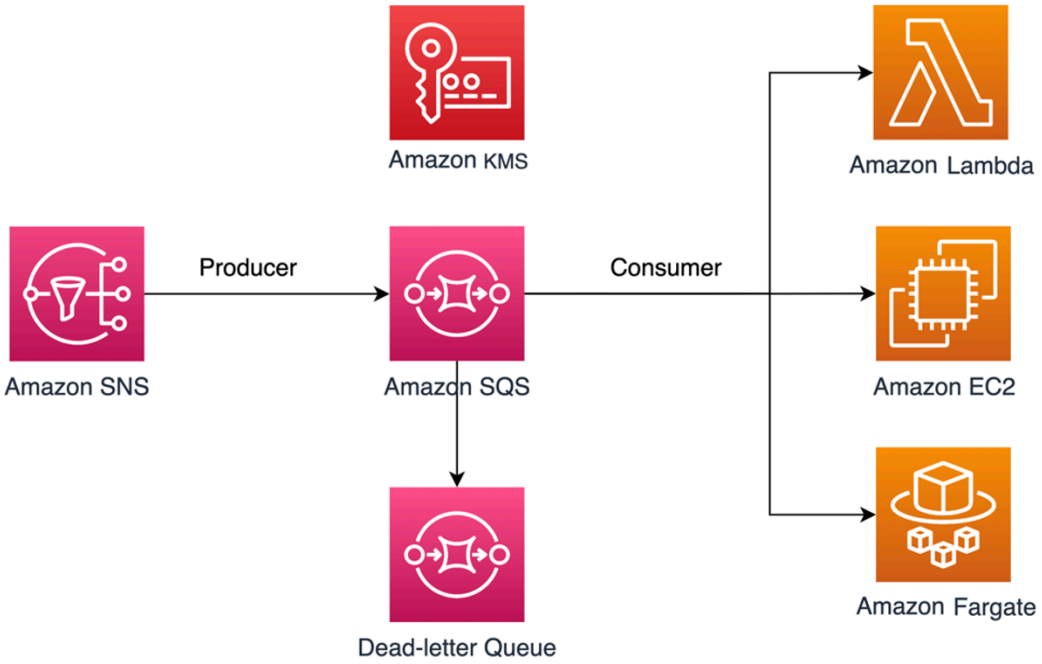
Questa guida descrive anche come le policy basate sulle risorse risolvono il problema confused-deputy utilizzando le chiavi di contesto delle condizioni IAM globali `aws:SourceArn`, `aws:SourceAccount` e `aws:PrincipalOrgID`.

Argomenti

- Panoramica
- Policy della chiave con privilegio minimo per Amazon SQS
- Dichiarazioni sulle policy di Amazon SQS per la coda DLQ
- Prevenzione del problema "confused deputy" tra servizi
- Utilizza IAM Access Analyzer per esaminare l'accesso multi-account

Panoramica

In questo argomento, descriveremo un caso d'uso comune per illustrare come creare la policy della chiave e la policy di coda di Amazon SQS. Questo caso d'uso viene mostrato nell'immagine seguente.



In questo esempio, il produttore del messaggio è un argomento di Amazon Simple Notification Service (SNS), configurato per separare i messaggi dalla coda crittografata di Amazon SQS. Il consumatore di messaggi è un servizio di elaborazione, come una funzione AWS Lambda, un'istanza Amazon Elastic Compute Cloud (EC2) o un container AWS Fargate. La coda Amazon SQS viene quindi configurata per inviare messaggi non riusciti a una coda DLQ. Le code DLQ sono utili per il debug di un'applicazione o di un sistema di messaggistica perché consentono di isolare messaggi non consumati e stabilire il motivo per cui la loro elaborazione non è riuscita. Nella soluzione definita in questo argomento, un servizio di elaborazione come una funzione Lambda viene utilizzato per elaborare i messaggi archiviati nella coda Amazon SQS. Se l'utente dei messaggi si trova in un cloud privato virtuale (VPC), l'istruzione della policy `DenyReceivingIfNotThroughVPCE` inclusa in questa guida consente di limitare la ricezione dei messaggi a quel VPC specifico.

> **Note**
>
> Questa guida contiene solo le autorizzazioni IAM richieste sotto forma di istruzioni di policy. Per creare la policy, devi aggiungere le istruzioni alla tua policy Amazon SQS o alla tua policy della chiave AWS KMS. Questa guida non fornisce istruzioni su come creare la coda Amazon

SQS o la chiave AWS KMS. Per istruzioni su come creare queste risorse, consulta Creazione di una coda Amazon SQS e Creazione di chiavi.
La policy di Amazon SQS definita in questa guida non supporta il reindirizzamento dei messaggi direttamente nella stessa coda Amazon SQS o in un'altra.

Policy della chiave con privilegio minimo per Amazon SQS

In questa sezione, descriviamo le autorizzazioni con privilegio minimo richieste in AWS KMS per la chiave gestita dal cliente che utilizzi per crittografare la tua coda Amazon SQS. Con queste autorizzazioni, puoi limitare l'accesso solo alle entità previste implementando il privilegio minimo. La policy della chiave deve contenere le seguenti istruzioni di policy, che descriveremo in dettaglio di seguito:

- Concessione di autorizzazioni ad amministratori per la chiave AWS KMS
- Concedi l'accesso in sola lettura ai metadati della chiave
- Concedi ad Amazon SNS le autorizzazioni KMS per pubblicare i messaggi nella coda
- Consenti ai consumatori di decrittografare i messaggi dalla coda

Concessione di autorizzazioni ad amministratori per la chiave AWS KMS

Per creare una chiave AWS KMS, devi fornire le autorizzazioni di amministratore AWS KMS per il ruolo IAM che utilizzi per implementare la chiave AWS KMS. Queste autorizzazioni di amministratore sono definite nella seguente istruzione di policy `AllowKeyAdminPermissions`. Quando aggiungi questa istruzione alla tua policy della chiave AWS KMS, assicurati di sostituire *<admin-role ARN>* con il nome della risorsa Amazon (ARN) del ruolo IAM utilizzato per implementare la chiave AWS KMS, gestire la chiave AWS KMS o entrambe le cose. Questo può essere il ruolo IAM della tua pipeline di implementazione o il ruolo di amministratore per la tua organizzazione nelle AWS Organizations.

```
{
  "Sid": "AllowKeyAdminPermissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "<admin-role ARN>"
    ]
  },
```

```
  "Action": [
    "kms:Create*",
    "kms:Describe*",
    "kms:Enable*",
    "kms:List*",
    "kms:Put*",
    "kms:Update*",
    "kms:Revoke*",
    "kms:Disable*",
    "kms:Get*",
    "kms:Delete*",
    "kms:TagResource",
    "kms:UntagResource",
    "kms:ScheduleKeyDeletion",
    "kms:CancelKeyDeletion"
  ],
  "Resource": "*"
}
```

> **Note**
>
> In una policy della chiave AWS KMS, il valore dell'elemento `Resource` deve essere `*`, il che significa “questa chiave AWS KMS“. L'asterisco (`*`) identifica la chiave AWS KMS a cui è collegata la policy della chiave.

Concedi l'accesso in sola lettura ai metadati della chiave

Per concedere ad altri ruoli IAM l'accesso in sola lettura ai metadati della chiave, aggiungi l'istruzione `AllowReadAccessToKeyMetaData` alla policy della tua chiave. Ad esempio, la seguente istruzione ti consente di elencare tutte le chiavi AWS KMS del tuo account a fini di controllo. Questa istruzione concede all'utente root AWS l'accesso in sola lettura ai metadati della chiave. Pertanto, qualsiasi principale IAM dell'account può avere accesso ai metadati della chiave se le relative policy basate sull'identità includono autorizzazioni elencate nella seguente istruzione: `kms:Describe*`, `kms:Get*` e `kms:List*`. Sostituisci *`<account-ID>`* con le tue informazioni.

```
{
  "Sid": "AllowReadAcesssToKeyMetaData",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
```

```
      "arn:aws:iam::<accountID>:root"
    ]
  },
  "Action": [
    "kms:Describe*",
    "kms:Get*",
    "kms:List*"
  ],
  "Resource": "*"
}
```

Concedi ad Amazon SNS le autorizzazioni KMS per pubblicare i messaggi nella coda

Per consentire al tuo argomento Amazon SNS di pubblicare messaggi nella coda crittografata di Amazon SQS, aggiungi l'informativa sulla policy `AllowSNSToSendToSQS` alla policy della chiave. Questa istruzione concede ad Amazon SNS l'autorizzazione a utilizzare la chiave AWS KMS per la pubblicazione nella coda Amazon SQS. Sostituisci *<account-ID>* con le tue informazioni.

> **Note**
>
> La `Condition` nell'istruzione limita l'accesso solo al servizio Amazon SNS nello stesso account AWS.

```
{
  "Sid": "AllowSNSToSendToSQS",
  "Effect": "Allow",
  "Principal": {
    "Service": [
      "sns.amazonaws.com"
    ]
  },
  "Action": [
    "kms:GenerateDataKey",
    "kms:Decrypt"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "aws:SourceAccount": "<account-id>"
    }
  }
```

```
}
```

Consenti ai consumatori di decrittografare i messaggi dalla coda

La seguente istruzione `AllowConsumersToReceiveFromTheQueue` concede all'utente di messaggi Amazon SQS le autorizzazioni necessarie per decrittografare i messaggi ricevuti dalla coda crittografata di Amazon SQS. Quando alleghi l'istruzione della policy, sostituisci *<consumer's runtime role ARN>* con l'ARN del ruolo di runtime IAM del consumatore del messaggio.

```
{
  "Sid": "AllowConsumersToReceiveFromTheQueue",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "<consumer's execution role ARN>"
    ]
  },
  "Action": [
    "kms:Decrypt"
  ],
  "Resource": "*"
}
```

Policy relativa al privilegio minimo per Amazon SQS

Questa sezione illustra le politiche di coda di Amazon SQS con privilegio minimo per il caso d'uso trattato in questa guida (ad esempio, da Amazon SNS ad Amazon SQS). La policy definita è progettata per prevenire accessi involontari utilizzando una combinazione delle istruzioni `Deny` e `Allow`. Le istruzioni `Allow` concedono l'accesso all'entità o alle entità previste. Le istruzioni `Deny` impediscono ad altre entità indesiderate di accedere alla coda di Amazon SQS, escludendo al contempo l'entità prevista dalla condizione della politica.

La policy di Amazon SQS include le seguenti istruzioni, che descriviamo in dettaglio di seguito:

- Limita le autorizzazioni di gestione di Amazon SQS
- Limita le azioni di coda di Amazon SQS dall'organizzazione specificata
- Concedi le autorizzazioni Amazon SQS ai consumatori
- Applica la crittografia in transito

- Limita la trasmissione di messaggi a un argomento specifico di Amazon SNS
- (Opzionale) Limitazione della ricezione dei messaggi a un endpoint VPC specifico

Limita le autorizzazioni di gestione di Amazon SQS

La seguente istruzione della policy `RestrictAdminQueueActions` limita le autorizzazioni di gestione di Amazon SQS solo al ruolo o ai ruoli IAM utilizzati per implementare la coda, gestire la coda o entrambe le cose. Sostituisci *<placeholder values>* con le tue informazioni. Specifica l'ARN del ruolo IAM utilizzato per implementare la coda Amazon SQS, nonché gli ARN di tutti i ruoli di amministratore che devono disporre delle autorizzazioni di gestione di Amazon SQS.

```
{
  "Sid": "RestrictAdminQueueActions",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": [
    "sqs:AddPermission",
    "sqs:DeleteQueue",
    "sqs:RemovePermission",
    "sqs:SetQueueAttributes"
  ],
  "Resource": "<SQS Queue ARN>",
  "Condition": {
    "StringNotLike": {
      "aws:PrincipalARN": [
        "arn:aws:iam::<account-id>:role/<deployment-role-name>",
        "<admin-role ARN>"
      ]
    }
  }
}
```

Limita le azioni di coda di Amazon SQS dall'organizzazione specificata

Per proteggere le tue risorse Amazon SQS dall'accesso esterno (accesso da parte di un'entità esterna all'organizzazione AWS), utilizza la seguente istruzione. Questa istruzione limita l'accesso alla coda di Amazon SQS all'organizzazione specificata in `Condition`. Assicurati di sostituire *<SQS queue ARN>* con l'ARN del ruolo IAM utilizzato per implementare la coda Amazon SQS e poi con l'ID dell'organizzazione *<org-id>*.

```
{
  "Sid": "DenyQueueActionsOutsideOrg",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": [
    "sqs:AddPermission",
    "sqs:ChangeMessageVisibility",
    "sqs:DeleteQueue",
    "sqs:RemovePermission",
    "sqs:SetQueueAttributes",
    "sqs:ReceiveMessage"
  ],
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "StringNotEquals": {
      "aws:PrincipalOrgID": [
        "<org-id>"
      ]
    }
  }
}
```

Concedi le autorizzazioni Amazon SQS ai consumatori

Per ricevere messaggi dalla coda Amazon SQS, devi fornire al consumatore dei messaggi le autorizzazioni necessarie. La seguente istruzione di policy concede al consumatore, da te specificato, le autorizzazioni necessarie per utilizzare i messaggi dalla coda di Amazon SQS. Quando aggiungi l'istruzione alla tua policy di Amazon SQS, assicurati di sostituire *<consumer's IAM runtime role ARN>* con l'ARN del ruolo di runtime IAM utilizzato dal consumatore e *<SQS queue ARN>* con l'ARN del ruolo IAM utilizzato per implementare la coda Amazon SQS.

```
{
  "Sid": "AllowConsumersToReceiveFromTheQueue",
  "Effect": "Allow",
  "Principal": {
    "AWS": "<consumer's IAM execution role ARN>"
  },
  "Action": [
    "sqs:ChangeMessageVisibility",
    "sqs:DeleteMessage",
```

```
    "sqs:GetQueueAttributes",
    "sqs:ReceiveMessage"
  ],
  "Resource": "<SQS queue ARN>"
}
```

Per impedire ad altre entità di ricevere messaggi dalla coda di Amazon SQS, aggiungi l'istruzione `DenyOtherConsumersFromReceiving` alla policy della coda di Amazon SQS. Questa istruzione limita il consumo di messaggi al consumatore specificato, senza consentire ad altri consumatori di accedervi, anche se le autorizzazioni di identità consentirebbero loro l'accesso. Assicurati di sostituire *<SQS queue ARN>* e *<consumer's runtime role ARN>* con le tue informazioni.

```
{
  "Sid": "DenyOtherConsumersFromReceiving",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": [
    "sqs:ChangeMessageVisibility",
    "sqs:DeleteMessage",
    "sqs:ReceiveMessage"
  ],
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "StringNotLike": {
      "aws:PrincipalARN": "<consumer's execution role ARN>"
    }
  }
}
```

Applica la crittografia in transito

La seguente istruzione di policy `DenyUnsecureTransport` impone a consumatori e produttori di utilizzare canali sicuri (connessioni TLS) per inviare e ricevere messaggi dalla coda di Amazon SQS. Assicurati di sostituire *<SQS queue ARN>* con l'ARN del ruolo IAM utilizzato per implementare la coda Amazon SQS.

```
{
```

```
  "Sid": "DenyUnsecureTransport",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": [
    "sqs:ReceiveMessage",
    "sqs:SendMessage"
  ],
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "Bool": {
      "aws:SecureTransport": "false"
    }
  }
}
```

Limita la trasmissione di messaggi a un argomento specifico di Amazon SNS

Il seguente è un esempio di istruzione di policy `AllowSNSToSendToTheQueue` che consente all'argomento Amazon SNS di inviare messaggi alla coda Amazon SQS. Assicurati di sostituire *<SQS queue ARN>* con l'ARN del ruolo IAM utilizzato per implementare la coda Amazon SQS e poi *<SNS topic ARN>* con l'ARN dell'argomento SNS.

```
{
  "Sid": "AllowSNSToSendToTheQueue",
  "Effect": "Allow",
  "Principal": {
    "Service": "sns.amazonaws.com"
  },
  "Action": "sqs:SendMessage",
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "ArnLike": {
      "aws:SourceArn": "<SNS topic ARN>"
    }
  }
}
```

La seguente istruzione di policy `DenyAllProducersExceptSNSFromSending` impedisce ad altri produttori di inviare messaggi alla coda. Sostituisci *<SQS queue ARN>* e *<SNS topic ARN>* con le tue informazioni.

```
{
  "Sid": "DenyAllProducersExceptSNSFromSending",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": "sqs:SendMessage",
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "ArnNotLike": {
      "aws:SourceArn": "<SNS topic ARN>"
    }
  }
}
```

(Opzionale) Limitazione della ricezione dei messaggi a un endpoint VPC specifico

Per limitare la ricezione di messaggi solo a un endpoint VPC specifico, aggiungi la seguente istruzione di policy alla tua policy di coda di Amazon SQS. Questa istruzione impedisce a un utente che consuma messaggi di ricevere messaggi dalla coda a meno che i messaggi non provengano dall'endpoint VPC desiderato. Sostituisci *<SQS queue ARN>* con l'ARN del ruolo IAM utilizzato per implementare la coda Amazon SQS e *<vpce_id>* con l'ID dell'endpoint VPC.

```
{
  "Sid": "DenyReceivingIfNotThroughVPCE",
  "Effect": "Deny",
  "Principal": "*",
  "Action": [
    "sqs:ReceiveMessage"
  ],
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "StringNotEquals": {
      "aws:sourceVpce": "<vpce id>"
    }
  }
```

```
}
```

Dichiarazioni sulle policy di Amazon SQS per la coda DLQ

Aggiungi le seguenti istruzioni di policy, identificate dal relativo ID, alla tua policy di accesso DLQ:

- `RestrictAdminQueueActions`
- `DenyQueueActionsOutsideOrg`
- `AllowConsumersToReceiveFromTheQueue`
- `DenyOtherConsumersFromReceiving`
- `DenyUnsecureTransport`

Oltre ad aggiungere le precedenti istruzioni di policy alla tua policy di accesso DLQ, devi aggiungere anche un'istruzione per limitare la trasmissione di messaggi alle code di Amazon SQS, come descritto nella sezione seguente.

Limitazione della trasmissione di messaggi a code Amazon SQS

Per limitare l'accesso solo alle code Amazon SQS dallo stesso account, aggiungi la seguente istruzione di policy `DenyAnyProducersExceptSQS` alla policy di coda DLQ. Questa istruzione non limita la trasmissione dei messaggi a una coda specifica perché prima di creare la coda principale devi implementare il DLQ, quindi non conoscerai l'ARN di Amazon SQS quando crei la DLQ. Se devi limitare l'accesso a una sola coda Amazon SQS, modifica la `aws:SourceArn` in `Condition` con l'ARN della tua coda di origine Amazon SQS, quando lo conosci.

```
{
  "Sid": "DenyAnyProducersExceptSQS",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": "sqs:SendMessage",
  "Resource": "<SQS DLQ ARN>",
  "Condition": {
    "ArnNotLike": {
      "aws:SourceArn": "arn:aws:sqs:<region>:<account-id>:*"
    }
  }
}
```

> **Important**
>
> Le policy di coda di Amazon SQS definite in questa guida non limitano l'operazione `sqs:PurgeQueue` a uno o più ruoli IAM specifici. L'azione `sqs:PurgeQueue` consente di eliminare tutti i messaggi nella coda Amazon SQS. Puoi anche utilizzare questa azione per apportare modifiche al formato del messaggio senza sostituire la coda Amazon SQS. Durante il debug di un'applicazione, puoi cancellare la coda di Amazon SQS per rimuovere messaggi potenzialmente errati. Durante il test dell'applicazione, puoi indirizzare un volume di messaggi elevato attraverso la coda di Amazon SQS e quindi eliminare la coda per ricominciare da capo prima di entrare in produzione. Il motivo per cui non si limita questa azione a un determinato ruolo è che questo ruolo potrebbe non essere noto durante l'implementazione della coda Amazon SQS. Dovrai aggiungere questa autorizzazione alla policy basata sull'identità del ruolo per poter eliminare la coda.

Prevenzione del problema "confused deputy" tra servizi

Il confused deputy è un problema di sicurezza in cui un'entità che non dispone dell'autorizzazione per eseguire un'azione può costringere un'entità dotata di privilegi maggiori a eseguire l'azione. Per evitarlo, AWS fornisce strumenti che ti aiutano a proteggere l'account se permetti a terze parti (accesso tra account) o ad altri servizi AWS (accesso tra servizi) di accedere a risorse nel tuo account. Le istruzioni di policy in questa sezione possono aiutarti a evitare il problema del "confused deputy" tra servizi.

La rappresentazione tra servizi può verificarsi quando un servizio (il servizio chiamante) effettua una chiamata a un altro servizio (il servizio chiamato). Il servizio chiamante può essere manipolato per utilizzare le proprie autorizzazioni e agire sulle risorse di un altro cliente, a cui normalmente non avrebbe accesso. Per contribuire alla protezione da questo problema, le policy basate sulle risorse definite in questo post utilizzano le chiavi di contesto delle condizioni IAM globali `aws:SourceArn`, `aws:SourceAccount` e `aws:PrincipalOrgID`. Ciò limita le autorizzazioni di cui dispone un servizio per una risorsa specifica, un account specifico o un'organizzazione specifica in AWS Organizations.

Utilizza IAM Access Analyzer per esaminare l'accesso multi-account

Utilizza AWS IAM Access Analyzer per esaminare le policy di coda Amazon SQS e le policy della chiave AWS KMS e ricevere avvisi quando una coda Amazon SQS o una chiave AWS KMS concede l'accesso a un'entità esterna. IAM Access Analyzer consente di identificare le risorse

nell'organizzazione e negli account che sono condivisi con un'entità esterna. Questa zona di fiducia può essere un account AWS o l'organizzazione all'interno di AWS Organizations specificata quando abiliti IAM Access Analyzer.

Sistema di analisi degli accessi AWS IAM individua le risorse condivise con entità esterne utilizzando il ragionamento basato sulla logica per analizzare le policy basate sulle risorse nell'ambiente AWS. Per ogni istanza di una risorsa condivisa al di fuori della zona di fiducia, Access Analyzer genera una ricerca. I risultati comprendono informazioni sull'accesso e sull'entità esterna a cui è concesso. È possibile rivedere i risultati per determinare se l'accesso è intenzionale e sicuro o se invece è involontario e rappresenta un rischio per la sicurezza. Per eventuali accessi non intenzionali, rivedi la policy interessata e correggila. Consulta questo post del blog per ulteriori informazioni su come AWS IAM Access Analyzer identifica gli accessi non intenzionali alle tue risorse AWS.

Per ulteriori informazioni su AWS IAM Access Analyzer, consulta la documentazione di AWS IAM Access Analyzer.

Autorizzazioni API Amazon SQS: riferimento a operazioni e risorse

Quando configuri Controllo accessi e scrivi policy di autorizzazione che è possibile collegare a un'identità IAM, utilizza la tabella seguente come riferimento. L'elenco include ogni operazione Amazon Simple Queue Service, le operazioni corrispondenti per le quali puoi concedere autorizzazioni di esecuzione e la risorsa AWS per la quale puoi concedere le autorizzazioni.

Puoi specificare le operazioni nel campo `Action` della policy e il valore della risorsa nel campo `Resource`. Per specificare un'operazione, utilizzare il prefisso `sqs:` seguito dal nome dell'azione (ad esempio `sqs:CreateQueue`).

Attualmente, Amazon SQS supporta le chiavi di contesto delle condizioni globali disponibili in IAM.

API di Amazon Simple Queue Service e autorizzazioni richieste per le azioni

AddPermission

- Operazioni: `sqs:AddPermission`
- Risorsa: `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

ChangeMessageVisibility

- Operazioni: `sqs:ChangeMessageVisibility`
- Risorsa: `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

ChangeMessageVisibilityBatch

Operazioni: `sqs:ChangeMessageVisibilityBatch`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

CreateQueue

Operazioni: `sqs:CreateQueue`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

DeleteMessage

Operazioni: `sqs:DeleteMessage`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

DeleteMessageBatch

Operazioni: `sqs:DeleteMessageBatch`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

DeleteQueue

Operazioni: `sqs:DeleteQueue`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

GetQueueAttributes

Operazioni: `sqs:GetQueueAttributes`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

GetQueueUrl

Operazioni: `sqs:GetQueueUrl`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

ListDeadLetterSourceQueues

Operazioni: `sqs:ListDeadLetterSourceQueues`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

ListQueues

Operazioni: `sqs:ListQueues`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

ListQueueTags

Operazioni: `sqs:ListQueueTags`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

PurgeQueue

Operazioni: `sqs:PurgeQueue`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

ReceiveMessage

Operazioni: `sqs:ReceiveMessage`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

RemovePermission

Operazioni: `sqs:RemovePermission`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

SendMessagee SendMessageBatch

Operazioni: `sqs:SendMessage`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

SetQueueAttributes

Operazioni: `sqs:SetQueueAttributes`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

TagQueue

Operazioni: `sqs:TagQueue`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

UntagQueue

Operazioni: `sqs:UntagQueue`

Risorsa: `arn:aws:sqs:region:account_id:queue_name`

Registrazione e monitoraggio in Amazon SQS

Questa sezione fornisce informazioni sul monitoraggio e la registrazione di code Amazon SQS.

Argomenti

- Registrazione delle chiamate API di Amazon SQS tramite AWS CloudTrail
- Monitoraggio delle code Amazon SQS tramite CloudWatch

Registrazione delle chiamate API di Amazon SQS tramite AWS CloudTrail

Amazon SQS è integrato AWS CloudTrail per registrare le chiamate Amazon SQS da un utente, ruolo o servizio. AWS CloudTrail acquisisce le chiamate API relative alle code standard e FIFO di Amazon SQS come eventi, incluse le interazioni avviate tramite la console Amazon SQS e programmaticamente tramite chiamate alle API di Amazon SQS.

Informazioni su Amazon SQS in CloudTrail

CloudTrail è attivata per impostazione predefinita quando crei il tuo AWS account. Quando si verifica un'attività di evento Amazon SQS supportata, viene registrata in un CloudTrail evento, insieme ad altri eventi di AWS servizio, nella cronologia degli eventi. Puoi visualizzare, ricercare e scaricare eventi recenti per l'account AWS. Per ulteriori informazioni, consulta Visualizzazione degli eventi con cronologia degli CloudTrail eventi nella Guida per l'AWS CloudTrailutente.

Le API di Amazon SQS che richiamano le operazioni di gestione delle code, ad esempio, `AddPermission` sono classificate come eventi di gestione e vengono registrate per impostazione predefinita. CloudTrail Le API di Amazon SQS che sono operazioni ad alto volume eseguite su una coda Amazon SQS, ad esempio, `SendMessage` sono classificate come eventi relativi ai dati e vengono registrate dopo l'attivazione. CloudTrail

Utilizzando le informazioni CloudTrail raccolte, puoi identificare una richiesta specifica a un'API Amazon SQS, l'indirizzo IP o l'identità del richiedente e la data e l'ora della richiesta. Se configuri un CloudTrail trail, puoi distribuire continuamente CloudTrail eventi a un bucket Amazon S3 con una

consegna opzionale ad Amazon CloudWatch Logs e. AWS EventBridge Se non configuri un trail, puoi solo visualizzare la cronologia degli eventi di gestione negli eventi nella console. CloudTrail Per ulteriori informazioni, consultare Panoramica per la creazione di un percorso nella Guida per l'utente di AWS CloudTrail.

Eventi di gestione in CloudTrail

Amazon SQS registra le seguenti azioni API come eventi di gestione:

- `AddPermission`
- `CreateQueue`
- `CancelMessageMoveTask`
- `DeleteQueue`
- `ListMessageMoveTasks`
- `PurgeQueue`
- `RemovePermission`
- `SetQueueAttributes`
- `StartMessageMoveTask`
- `TagQueue`
- `UntagQueue`

Le seguenti API Amazon SQS non sono supportate per la registrazione: CloudTrail

- `GetQueueAttributes`
- `GetQueueUrl`
- `ListDeadLetterSourceQueues`
- `ListQueueTags`
- `ListQueues`

Eventi relativi ai dati in CloudTrail

Gli eventi di dati forniscono informazioni sulle operazioni eseguite su o in una risorsa, come l'invio o la ricezione di un messaggio Amazon SQS da e verso una coda Amazon SQS. Gli eventi relativi ai dati sono attività ad alto volume che CloudTrail non vengono registrate per impostazione predefinita.

Puoi abilitare la registrazione delle azioni dell'API degli eventi relativi ai dati per la tua coda SQS utilizzando le API. CloudTrail Per ulteriori informazioni, consultare Registrazione di eventi di dati nella Guida per l'utente di AWS CloudTrail.

Con CloudTrail, puoi utilizzare selettori di eventi avanzati per decidere quali attività dell'API Amazon SQS vengono registrate e registrate. Per effettuare il logging di eventi di dati di Amazon SQS, devi includere il tipo di risorsa `AWS::SQS::Queue`. Una volta impostato il tipo di risorsa, puoi affinare ulteriormente le tue preferenze di logging selezionando eventi di dati specifici da registrare, ad esempio utilizzando il filtro `eventName` per tenere traccia degli eventi `SendMessage`. Per ulteriori informazioni, consulta `AdvancedEventSelector` nella documentazione di riferimento dell'API AWS CloudTrail.

Eventi di dati di Amazon SQS:

- `SendMessage`
- `SendMessageBatch`
- `ReceiveMessage`
- `DeleteMessage`
- `DeleteMessageBatch`
- `ChangeMessageVisibility`
- `ChangeMessageVisibilityBatch`

Per gli eventi di dati sono previsti costi aggiuntivi. Per ulteriori informazioni, consulta la pagina Prezzi. AWS CloudTrail

Esempi: eventi CloudTrail di gestione per Amazon SQS

Gli esempi seguenti mostrano le voci di CloudTrail log per le API supportate:

AddPermission

L'esempio seguente mostra una voce di CloudTrail registro per una chiamata `AddPermission` API.

```
{
   "Records": [
     {
       "eventVersion": "1.06",
       "userIdentity": {
         "type": "IAMUser",
```

```
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/Alice",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Alice"
      },
      "eventTime": "2018-06-28T22:23:46Z",
      "eventSource": "sqs.amazonaws.com",
      "eventName": "AddPermission",
      "awsRegion": "us-east-2",
      "sourceIPAddress": "203.0.113.0",
      "userAgent": "Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20100101
 Firefox/24.0",
      "requestParameters": {
        "actions": [
          "SendMessage"
        ],
        "AWSAccountIds": [
          "123456789012"
        ],
        "label": "MyLabel",
        "queueUrl": "https://sqs.us-east-2.amazon.com/123456789012/MyQueue"
      },
      "responseElements": null,
      "requestID": "123abcde-f4gh-50ij-klmn-60o789012p30",
      "eventID": "0987g654-32f1-09e8-d765-c4f3fb2109fa"
    }
  ]
}
```

CreateQueue

L'esempio seguente mostra una voce di CloudTrail registro per una chiamata `CreateQueue` API.

```
{
   "Records": [
     {
       "eventVersion": "1.06",
       "userIdentity": {
         "type": "IAMUser",
         "principalId": "AKIAI44QH8DHBEXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/Alejandro",
         "accountId": "123456789012",
         "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
```

```
        "userName": "Alejandro"
      },
      "eventTime": "2018-06-28T22:23:46Z",
      "eventSource": "sqs.amazonaws.com",
      "eventName": "CreateQueue",
      "awsRegion": "us-east-2",
      "sourceIPAddress": "203.0.113.1",
      "userAgent": "Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20100101
 Firefox/24.0",
      "requestParameters": {
        "queueName": "MyQueue"
      },
      "responseElements": {
        "queueUrl": "https://sqs.us-east-2.amazon.com/123456789012/MyQueue"
      },
      "requestID": "123abcde-f4gh-50ij-klmn-60o789012p30",
      "eventID": "0987g654-32f1-09e8-d765-c4f3fb2109fa"
    }
  ]
}
```

DeleteQueue

L'esempio seguente mostra una voce di CloudTrail registro per una chiamata `DeleteQueue` API.

```
{
   "Records": [
     {
       "eventVersion": "1.06",
       "userIdentity": {
         "type": "IAMUser",
         "principalId": "AKIAI44QH8DHBEXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/Carlos",
         "accountId": "123456789012",
         "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
         "userName": "Carlos"
       },
       "eventTime": "2018-06-28T22:23:46Z",
       "eventSource": "sqs.amazonaws.com",
       "eventName": "DeleteQueue",
       "awsRegion": "us-east-2",
       "sourceIPAddress": "203.0.113.2",
       "userAgent": "Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20100101
 Firefox/24.0",
```

```
      "requestParameters": {
        "queueUrl": "https://sqs.us-east-2.amazon.com/123456789012/MyQueue"
      },
      "responseElements": null,
      "requestID": "123abcde-f4gh-50ij-klmn-60o789012p30",
      "eventID": "0987g654-32f1-09e8-d765-c4f3fb2109fa"
    }
  ]
}
```

RemovePermission

L'esempio seguente mostra una voce di CloudTrail registro per una chiamata `RemovePermission` API.

```
{
   "Records": [
     {
       "eventVersion": "1.06",
       "userIdentity": {
         "type": "IAMUser",
         "principalId": "AKIAI44QH8DHBEXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/Jane",
         "accountId": "123456789012",
         "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
         "userName": "Jane"
       },
       "eventTime": "2018-06-28T22:23:46Z",
       "eventSource": "sqs.amazonaws.com",
       "eventName": "RemovePermission",
       "awsRegion": "us-east-2",
       "sourceIPAddress": "203.0.113.3",
       "userAgent": "Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20100101
Firefox/24.0",
       "requestParameters": {
         "label": "label",
         "queueUrl": "https://sqs.us-east-2.amazon.com/123456789012/MyQueue"
       },
       "responseElements": null,
       "requestID": "123abcde-f4gh-50ij-klmn-60o789012p30",
       "eventID": "0987g654-32f1-09e8-d765-c4f3fb2109fa"
     }
   ]
```

```
 }
```

SetQueueAttributes

L'esempio seguente mostra una voce di CloudTrail registro per`SetQueueAttributes`:

```
{
   "Records": [
     {
       "eventVersion": "1.06",
       "userIdentity": {
         "type": "IAMUser",
         "principalId": "AKIAI44QH8DHBEXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/Maria",
         "accountId": "123456789012",
         "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
         "userName": "Maria"
       },
       "eventTime": "2018-06-28T22:23:46Z",
       "eventSource": "sqs.amazonaws.com",
       "eventName": "SetQueueAttributes",
       "awsRegion": "us-east-2",
       "sourceIPAddress": "203.0.113.4",
       "userAgent": "Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20100101
 Firefox/24.0",
       "requestParameters": {
         "attributes": {
           "VisibilityTimeout": "100"
         },
         "queueUrl": "https://sqs.us-east-2.amazon.com/123456789012/MyQueue"
       },
       "responseElements": null,
       "requestID": "123abcde-f4gh-50ij-klmn-60o789012p30",
       "eventID": "0987g654-32f1-09e8-d765-c4f3fb2109fa"
     }
   ]
 }
```

Esempi: eventi CloudTrail relativi ai dati per Amazon SQS

Di seguito sono riportati alcuni esempi di CloudTrail eventi specifici delle API per eventi di dati di Amazon SQS:

SendMessage

L'esempio seguente mostra un evento di CloudTrail dati per. `SendMessage`


```
    {
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
      },
      "attributes": {
        "creationDate": "2023-11-07T22:13:06Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:59:11Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "SendMessage",
  "awsRegion": "ap-southeast-4",
  "sourceIPAddress": "10.0.118.80",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue",
    "messageBody": "HIDDEN_DUE_TO_SECURITY_REASONS",
    "messageDeduplicationId": "MsgDedupIdSdk1ae1958f2-bbe8-4442-83e7-4916e3b035aa",
    "messageGroupId": "MsgGroupIdSdk16"
  },
  "responseElements": {
    "mD5OfMessageBody": "9a4e3f7a614d9dd9f8722092dbda17a2",
    "mD5OfMessageSystemAttributes": "f88f0587f951b7f5551f18ae699c3a9d",
```

```
    "messageId": "93bb6e2d-1090-416c-81b0-31eb1faa8cd8",
    "sequenceNumber": "18881790870905840128"
  },
  "requestID": "c4584600-fe8a-5aa3-a5ba-1bc42f055fae",
  "eventID": "98c735d8-70e0-4644-9432-b6ced4d791b1",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::SQS::Queue",
      "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "123456789012",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
  }
```

SendMessageBatch

L'esempio seguente mostra un evento di CloudTrail dati per`SendMessageBatch`.

```
    {
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
```

```
      },
      "attributes": {
        "creationDate": "2023-11-07T22:13:06Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:59:05Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "SendMessageBatch",
  "awsRegion": "ap-southeast-4",
  "sourceIPAddress": "10.0.118.80",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "delaySeconds": 0,
    "entries": [
      {
        "id": "0",
        "messageBody": "HIDDEN_DUE_TO_SECURITY_REASONS",
        "messageAttributes": [
          {
            "name": "HIDDEN_DUE_TO_SECURITY_REASONS"
          }
        ],
        "messageDeduplicationId": "MsgDedupIdSdk1027092b6-b6f6-41af-a084-e72d572a6d4b",
        "messageGroupId": "MsgGroupIdSdk12"
      }
    ],
    "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue"
  },
  "responseElements": {
    "successful": [
      {
        "id": "0",
        "messageId": "9048ab28-e38d-46da-b9fe-f70b3873f888",
        "mD5OfMessageBody": "0f1a575a56eb5cf5072a8dedc585d2dd",
        "mD5OfMessageAttributes": "6e1d6d5d774a05efe9df5eb000639db7",
        "sequenceNumber": "18881790869375471872",
        "mD5OfMessageSystemAttributes": "6f540b6e375dcda1aad2d4aaff28ebf8"
      }
    ]
  },
```

```
  "requestID": "b5a386a4-2d4a-5de3-9910-db60fcc368ee",
  "eventID": "20f5ecbe-2b0b-4d0b-a6f7-365bc94c4ca5",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue",
      "type": "AWS::SQS::Queue"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "123456789012",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
  }
}
```

ReceiveMessage

L'esempio seguente mostra un evento di CloudTrail dati per`ReceiveMessage`.

```
    {
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
      },
      "attributes": {
```

```
        "creationDate": "2023-11-07T22:13:06Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:59:24Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "ReceiveMessage",
  "awsRegion": "ap-southeast-4",
  "sourceIPAddress": "10.0.118.80",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue",
    "maxNumberOfMessages": 10
  },
  "responseElements": null,
  "requestID": "8b4d4643-8f49-52cd-a6e8-1b875ed54b99",
  "eventID": "f3f23ab7-b0a4-4b71-afc0-141209c49206",
  "readOnly": true,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::SQS::Queue",
      "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "123456789012",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
  }
}
```


DeleteMessage

L'esempio seguente mostra un evento di CloudTrail dati per`DeleteMessage`.

```
    {
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
      },
      "attributes": {
        "creationDate": "2023-11-07T22:13:06Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:58:42Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "DeleteMessage",
  "awsRegion": "ap-southeast-4",
  "sourceIPAddress": "10.0.118.80",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "receiptHandle": "AQEBfgYUKTy3dyOAewC4wI3lQZEB3oUDuv8M8FWhObnr3lqRsFBiZ57mmxO1/
dWfdlvGgDW7sRSry6HHxWrNfHItQMUHtWX3a/vEjJ6sWC/5Mf36I/B2HBLCT2zGO/
IZTywxFmUT4HUudWKcGpuZb/Kcl3Fom6hYU8PxxzPxLOKPtFwrVU+G2Spvf/
Tbuyj27h5+AkNxfaAhu/dnvXnAJcDJErGsJTjSS1i6iRzFq+jg6K5Fw6T578QJZcx/
ZLaCyohmj2HaOOktwhbqQc4j+2gKSfxrACgXCu6De5bCtwgtGdhMEh4DtVIQh88qGUCaofQ3t/
eRBIvIFJIa61JCVNWSBqOtELEIfxaHpSvo0c1IEecKDt1IJ08Cij3euLFMIzmUot24IViZt8ntKVAZ6KBLlLedrVlxOhNVc
WG0jfbqz3iBS1T1AD1zJKT7ICIA+edgaYJpOZw4=",
    "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue"
  },
  "responseElements": null,
  "requestID": "fbd23ff4-a107-536d-8fcb-623070754bc0",
```

```
  "eventID": "9951fed7-365f-4046-bc71-e5bf065a9b47",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::SQS::Queue",
      "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "123456789012",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
  }
}
```

DeleteMessageBatch

L'esempio seguente mostra un evento di CloudTrail dati per`DeleteMessageBatch`.

```
    {
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
      },
      "attributes": {
        "creationDate": "2023-11-07T22:13:06Z",
```

```
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:59:24Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "DeleteMessageBatch",
  "awsRegion": "ap-southeast-4",
  "sourceIPAddress": "10.0.118.80",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue",
    "entries": [
      {
        "id": "0",
        "receiptHandle": "AQEBefxM1O4zyZGF87DehbRbmri91w2W7mMdD0GrBjQa8e/
hpb4RbXHPZ9tLBVleECbChQIE5NtaDuoZhZPOkTy0eN46EyRR4jXDzE3AlkbPlX1mA9f2fUuTrXx8aeCoCA3I3woNg3fXXA
hlLS94tjAZqV2krc4BaC2pYgjyHWcW019HwIV8T/bjNMIeZoQwOM5V
+o9vHPfewz5QGr5SKpDo7uE7Umyk5n5CJZvcn1efp/
mrwtaCIb9M7cCQUYcZm2ZmZDnIO9XpGTAi3m2dQ0M83pnNh0nvDfpkHpoa+hX1TrUmxCupCWHJwA8HFJ1O/
CCJsodMNFthLBA9S57dkBZCsw41G8jAmgQ0MkvZ0UL5mg0OFQQd1Yrw0zvthjCgiwdzn0yXoMzxIZMBxkY14E4nVVZ7N5XE
h8oRk2C7gByzg2kYJ0LnUvLJFT8DQE28JZppEC9klvrdR/BWiPT7asc="
      }
    ]
  },
  "responseElements": {
    "successful": [
      {
        "id": "0"
      }
    ],
    "failed": []
  },
  "requestID": "fe423091-5642-5ba5-9256-6d5587de52f1",
  "eventID": "88c8020d-d769-4985-8ecb-ee0b59acc418",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::SQS::Queue",
      "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
    }
```

```
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "123456789012",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
  }
}
```

ChangeMessageVisibility

L'esempio seguente mostra un evento di CloudTrail dati per`ChangeMessageVisibility`.

```
    {
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
      },
      "attributes": {
        "creationDate": "2023-11-07T22:13:06Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:59:24Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "ChangeMessageVisibility",
  "awsRegion": "ap-southeast-4",
```

```
  "sourceIPAddress": "10.0.118.80",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue",
    "receiptHandle": "AQEBy+2qnmQQVxcXrEwN7t6dXkjGAllr5DuSpGlvHx9s/vwbwp+RIr3dD6vRvlsU/
lteIulKHBs6DEIR7KL+J3mACfB+RRpRlWPlguiCdLKNKSVpdhkSBDDvkRHfycTHjuszGIebGdl+tYYjPrlz
+DSePmpty0EdhqtorW1xAc0Xf0GZbt0FtkbRFK3ql5lETIHgthBCABoxuOCNvMElz9rYQ9m5OY30Z5YOZvQ/
coPHYl+9HhNV/A6Fs+/d6mVx9v6TomTh5L03wXqtjA8b0gkGftclQh/tJBAxqY/S8YG9OKtY4NDP0SQBtYF/
vCCsCq9+5fiUfiYyvtdHSlwP9AyRotenCGrUKaRFiRhxDm1D6up0UaBs2d8wgHdKFf/5mENTdeqrXQdZfwkFazW1a8ifWJm
+HzhfA9EJcdgWSS72WCMaerydsCxaX+E08B2ubL6oiafMYW4gK0GIRxYZ0+eeXKWy4TxkReW3j7k=",
    "visibilityTimeout": 1272
  },
  "responseElements": null,
  "requestID": "6fbefbde-55d9-5640-98d1-a61a84457f14",
  "eventID": "72275c61-bfc0-4606-934b-a6b7397aef20",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::SQS::Queue",
      "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "123456789012",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
  }
}
```


ChangeMessageVisibilityBatch

L'esempio seguente mostra un evento di CloudTrail dati per`ChangeMessageVisibilityBatch`.


```
    {
  "eventVersion": "1.09",
```

```
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
      },
      "attributes": {
        "creationDate": "2023-11-07T22:13:06Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:59:01Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "ChangeMessageVisibilityBatch",
  "awsRegion": "ap-southeast-4",
  "sourceIPAddress": "10.0.118.80",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "visibilityTimeout": 0,
    "entries": [
      {
        "id": "0",
        "receiptHandle":
 "AQEB2M5cVYg5gslhWME6537hdjcaPnOYPA5M0W460TTb0DzPle631yPWm8qxd4O1hDj/
B4ntTMnsgBTa95t14tNx7Vn96jKJ5rIoZ7iI8TRmkT1caKodKIPs8w9yndZq50c2FPQxtyH+2L3UHf/
abV3szqVWXOLZR4PwX8zZkWVQGNCNnY2q2lGCG586F8QwvrOFYoXNwB8ymd1t77e1PDPknq1Io3JFuzkEsndkkETy4fV1Qq
l5PHX17nXxaC+DURVlMPXOuSFACGmWqAoyk50HKwGOjLQgpySL/
TcnQXClvFq8kNXGwyVzJsbwHpOHxI7oce69vaD6DaWFP75d3hx+PJeG9pauQCKzVP3skt3Hw/
zDC7YfKcALD3aCwMmeNDwT3w0BUG6XZdG5lYhtFtTQYV7YuS3i/
Jh3HShGbtm07JKOEFiPkxv2+XNaAX3gFEpbng6zamTanfyMXCJIiglAEqiyWHQ=",
        "visibilityTimeout": 2271
      }
    ],
```

```
    "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue"
  },
  "responseElements": {
    "successful": [
      {
        "id": "0"
      }
    ]
  },
  "requestID": "d49ab65f-9dc7-54b8-875c-eb9b4c42988b",
  "eventID": "ca16c8c2-c4ba-4eb5-a54c-e650a10266d4",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::SQS::Queue",
      "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "123456789012",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
  }
}
```


Monitoraggio delle code Amazon SQS tramite CloudWatch

Amazon SQS e Amazon CloudWatch sono integrati, quindi puoi utilizzarli CloudWatch per visualizzare e analizzare i parametri per le tue code Amazon SQS. Puoi visualizzare e analizzare i parametri delle code dalla console Amazon SQS, dalla console, utilizzando o utilizzando l'API AWS CLI. CloudWatch CloudWatch Puoi anche impostare CloudWatch allarmi per i parametri di Amazon SQS.

CloudWatch le metriche per le code Amazon SQS vengono raccolte automaticamente e inserite a intervalli di un minuto. CloudWatch Queste metriche vengono raccolte su tutte le code che

soddisfano le linee guida per essere attive. CloudWatch CloudWatchconsidera attiva una coda per un massimo di sei ore se contiene messaggi o se un'azione vi accede.

Quando una coda Amazon SQS rimane inattiva per più di sei ore, il servizio Amazon SQS viene considerato inattivo e smette di fornire i parametri al servizio. CloudWatch I dati mancanti, o i dati che rappresentano zero, non possono essere visualizzati nelle CloudWatch metriche di Amazon SQS per il periodo di tempo in cui la coda Amazon SQS era inattiva.

> **Note**
>
> - Quando una coda viene attivata da uno stato inattivo, si verifica un ritardo fino a 15 minuti nelle CloudWatch metriche.
> - Non sono previsti costi per i parametri di Amazon SQS riportati in. CloudWatch Sono fornite come parte del servizio Amazon SQS.
> - CloudWatch le metriche sono supportate sia per le code standard che per quelle FIFO.

Argomenti

Accesso ai CloudWatch parametri per Amazon SQS

Amazon SQS e Amazon CloudWatch sono integrati, quindi puoi utilizzarli CloudWatch per visualizzare e analizzare i parametri per le tue code Amazon SQS. Puoi visualizzare e analizzare i parametri delle code dalla console Amazon SQS, dalla console, utilizzando o utilizzando l'API AWS CLI. CloudWatch CloudWatch Puoi anche impostare CloudWatch allarmi per i parametri di Amazon SQS.

Console Amazon SQS

1. Accedere alla console Amazon SQS.
2. Nell'elenco delle code, sceglie (seleziona) le caselle per le code ai cui parametri desideri accedere. Puoi visualizzare i parametri per un massimo di 10 code.
3. Scegliere la scheda Monitoring (Monitoraggio).

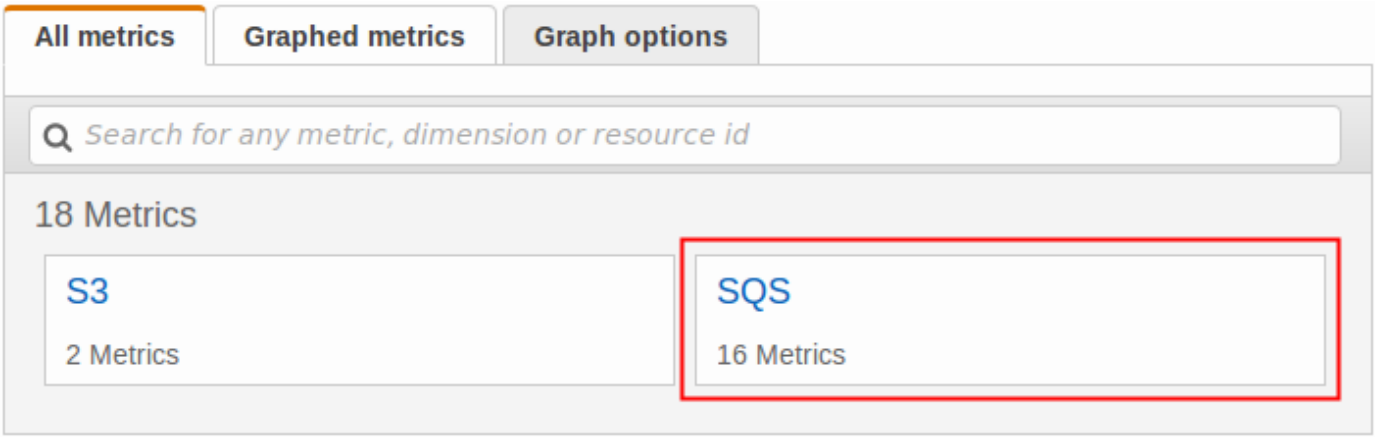
Nella sezione relativa ai parametri SQS vengono visualizzati diversi grafici.

4. Per comprendere ciò che un determinato grafico rappresenta, passa il mouse su

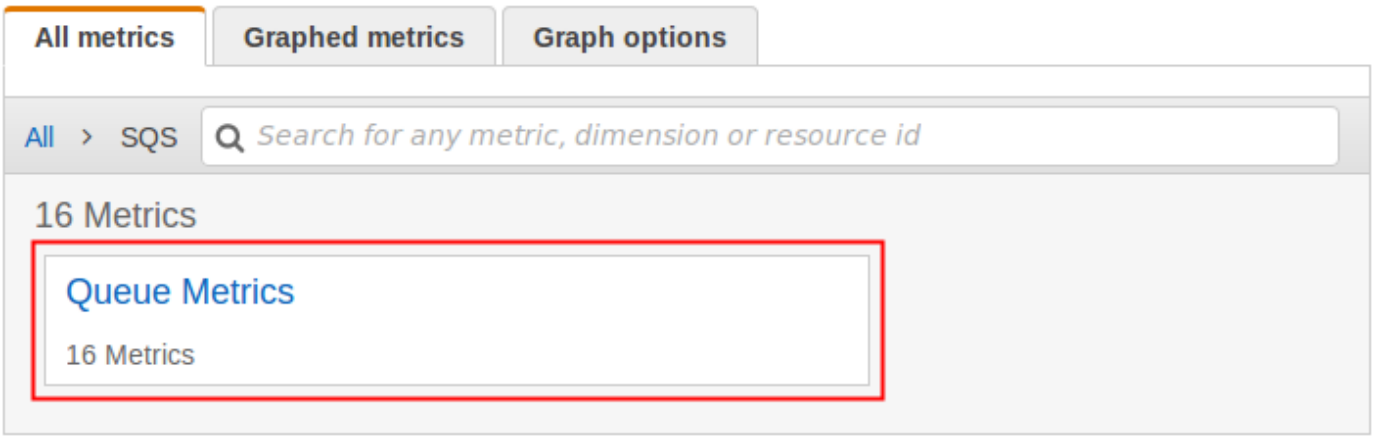
accanto al grafico desiderato, oppure consulta CloudWatch Metriche disponibili per Amazon SQS.
5. Per modificare l'intervallo di tempo per tutti i grafici contemporaneamente, per Time Range (Intervallo di tempo) scegli l'intervallo di tempo desiderato (per esempio, Last Hour (Ultima ora)).
6. Per visualizzare statistiche aggiuntive per un singolo grafico, scegli il grafico.
7. Nella finestra di dialogo Monitoring Details (Dettagli monitoraggio) CloudWatch , seleziona una Statistic (Statistica), (ad esempio, Sum (Somma)). Per un elenco di statistiche supportate, consulta CloudWatch Metriche disponibili per Amazon SQS.
8. Per modificare l'intervallo di tempo o il periodo di tempo per il quale un singolo grafico viene visualizzato (ad esempio, per mostrare un intervallo di tempo delle ultime 24 ore anziché gli ultimi 5 minuti, o per visualizzare un periodo di tempo di ogni ora anziché ogni 5 minuti), con la finestra di dialogo del grafico ancora visualizzata, per Time Range (Intervallo di tempo) scegli l'intervallo di tempo desiderato (per esempio, Last 24 Hours (Ultime 24 ore)). Per Period (Periodo) sceglie il periodo di tempo desiderato entro l'intervallo di tempo specificato (ad esempio, 1 Hour (1 ora)). Dopo aver finito di esaminare il grafico, scegli Close (Chiudi).
9. (Facoltativo) Per utilizzare CloudWatch funzionalità aggiuntive, nella scheda Monitoraggio, scegli Visualizza tutti i CloudWatch parametri, quindi segui le istruzioni della procedura. CloudWatch Console Amazon

CloudWatch Console Amazon

1. Accedi alla console CloudWatch.
2. Nel pannello di navigazione, scegli Metrics (Parametri).
3. Seleziona namespace parametro SQS.

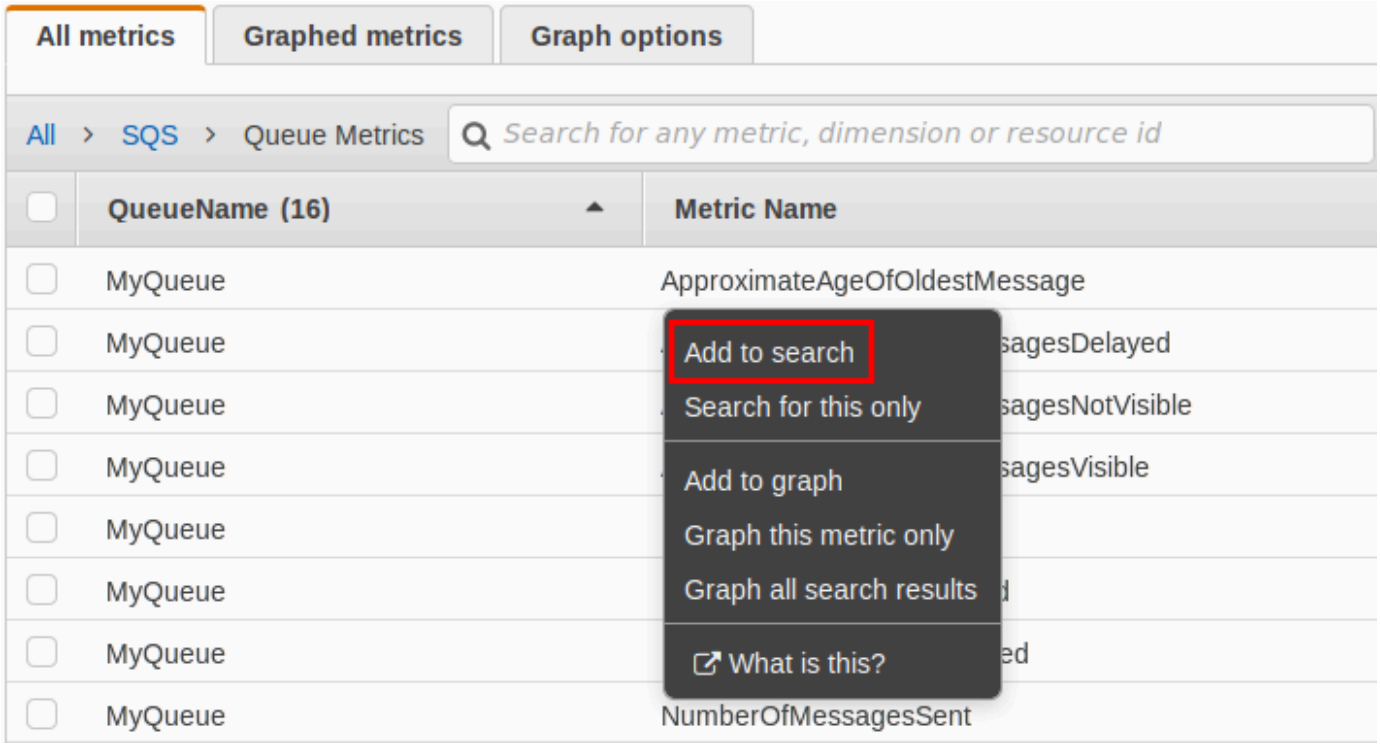


4. Seleziona la dimensione parametro Queue Metrics (parametri coda).



5. Puoi ora quindi esaminare le metriche Amazon SQS:

   - Per ordinare i parametri, utilizza l'intestazione della colonna.
   - Per creare il grafico di un parametro, seleziona la casella di controllo accanto al parametro.
   - Per filtrare in base a un parametro, scegli il nome del parametro e quindi Add to search (Aggiungi alla ricerca).



Per ulteriori informazioni e opzioni aggiuntive, consulta Graph Metrics e Using Amazon CloudWatch Dashboards nella Amazon CloudWatch User Guide.

AWS Command Line Interface

Per accedere ai parametri Amazon SQS utilizzando la AWS CLI, esegui il comando `get-metric-statistics`.

Per ulteriori informazioni, consulta Get Statistics for a Metric nella Amazon CloudWatch User Guide.

CloudWatch API

Per accedere ai parametri di Amazon SQS utilizzando l' CloudWatch API, utilizza l'azione. `GetMetricStatistics`

Per ulteriori informazioni, consulta Get Statistics for a Metric nella Amazon CloudWatch User Guide.

Creazione di CloudWatch allarmi per i parametri di Amazon SQS

CloudWatch consente di attivare allarmi in base a una soglia metrica. Ad esempio, puoi creare un allarme per il parametro `NumberOfMessagesSent`. Ad esempio, se vengono inviati più di

100 messaggi alla coda `MyQueue` in 1 ora, viene inviata una notifica tramite e-mail. Per ulteriori informazioni, consulta Creating Amazon CloudWatch Alarms nella Amazon CloudWatch User Guide.

1. Accedi AWS Management Console e apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Seleziona Alarms (Allarmi), quindi Create Alarm (Crea allarme).
3. Nella sezione Select Metric (Seleziona parametro) della finestra di dialogo Create Alarm (Crea allarme), scegli Browse Metrics (Sfoglia parametri), SQS.
4. Per SQS > Queue Metrics, scegli il nome QueueNamee la metrica per cui impostare un allarme, quindi scegli Avanti. Per un elenco di parametri disponibili, consulta CloudWatch Metriche disponibili per Amazon SQS.

   Nel seguente esempio, la selezione è per un allarme per il parametro `NumberOfMessagesSent` per la coda `MyQueue`. I trigger di allarme quando il numero di messaggi inviati supera 100.
5. Nella sezione Define Alarm (Definisci allarme) della finestra di dialogo Create Alarm (Crea allarme), procedi come segue:

   a. In Alarm Threshold (Soglia allarme), digita Name (Nome) e Description (Descrizione) per l'allarme.

   b. Imposta is su > 100.

   c. Imposta for (per) su 1 out of 1 datapoints (1 di 1 punto dati).

   d. In Alarm preview (Anteprima allarme), imposta Period (Periodo) su 1 Hour (1 ora).

   e. Imposta Statistic (Statistica) su Standard, Sum (Somma).

   f. In Actions (Operazioni), imposta Whenever this alarm (Ogni volta che l'allarme) su State is ALARM (Stato è ALLARME).

      Se desideri CloudWatch inviare una notifica quando viene attivato l'allarme, seleziona un argomento Amazon SNS esistente o scegli Nuovo elenco e inserisci gli indirizzi e-mail separati da virgole.

      > **Note**
      >
      > Se crei un nuovo argomento Amazon SNS, gli indirizzi e-mail devono essere verificati prima che possano ricevere le notifiche. Se lo stato dell'allarme cambia prima della verifica degli indirizzi e-mail, le notifiche non vengono inviate.

6. Scegli Crea allarme.

   L'allarme viene creato.

CloudWatch Metriche disponibili per Amazon SQS

Amazon SQS invia le seguenti metriche a. CloudWatch

> **Note**
>
> Per le code standard, il risultato è approssimativo a causa dell'architettura distribuita di Amazon SQS. Nella maggior parte dei casi, il conteggio deve essere vicino al numero effettivo di messaggi in coda.
> Per le code FIFO il risultato è esatto.

Metriche Amazon SQS

Lo spazio dei nomi `AWS/SQS` include le metriche descritte di seguito.

Metrica	Descrizione
`ApproximateAgeOfOldestMessage`	L'età approssimativa del messaggio non eliminato meno recente in coda. **Note** • Dopo che il messaggio viene ricevuto tre volte (o più) e non viene elaborato, il messaggio viene spostato alla fine della coda e il parametro `ApproximateAgeOfOldestMessage` punta al secondo messaggio meno recente che non è stato ricevuto più di tre volte. Questa

Metrica	Descrizione
	operazione si verifica anche se la coda dispone di una policy di reindirizzamento. • Poiché un singolo messaggio poison-pill (ricevuto più volte ma mai eliminato) può distorcere questo parametro, la durata di un messaggio poison-pill non viene inclusa nel parametro finché il messaggio poison-pill non viene elaborato. • Quando la coda ha una policy di reindirizzamento, il messaggio viene spostato in una coda DLQ dopo che il numero massimo configurato di ricezioni. Quando il messaggio viene spostato nella coda DLQ, il parametro `ApproximateAgeOfOldestMessage` della coda DLQ rappresenta l'ora in cui il messaggio è stato spostato nella coda DLQ (non l'ora originale in cui è stato inviato il messaggio). • Per le code FIFO, il messaggio non viene spostato in fondo alla coda perché ciò violerebbe la garanzia dell'ordine FIFO. Il messaggio verrà invece inviato al DLQ, se ne è stato configurato uno. Altrimenti bloccherà il

Metrica	Descrizione
	gruppo di messaggi fino alla sua eliminazione o fino alla scadenza. Criteri di segnalazione: se la coda è attiva, viene riportato un valore non negativo. Unità: secondi Statistiche valide: media, minimo, massimo, somma, esempi di dati (visualizzato come numero di esempi nella console di Amazon SQS)
`ApproximateNumberOfMessagesDelayed`	Il numero dei messaggi nella coda che vengono differiti e non sono disponibili per la lettura immediata. Ciò può accadere quando la coda è configurata come coda di ritardo o quando un messaggio è stato inviato con un parametro di ritardo. Criteri di report: se la coda è attiva, viene riportato un valore non negativo. Unità: numero Statistiche valide: media, minimo, massimo, somma, esempi di dati (visualizzato come numero di esempi nella console di Amazon SQS)

Metrica	Descrizione
`ApproximateNumberOfMessagesNotVisible`	Il numero di messaggi che sono in transito. I messaggi sono considerati in transito se sono stati inviati a un client ma non sono ancora stati eliminati o non hanno ancora raggiunto il termine della loro finestra di visibilità. Criteri di report: se la coda è attiva, viene riportato un valore non negativo. Unità: numero Statistiche valide: media, minimo, massimo, somma, esempi di dati (visualizzato come numero di esempi nella console di Amazon SQS)
`ApproximateNumberOfMessagesVisible`	Il numero dei messaggi da elaborare. Criteri di report: se la coda è attiva, viene riportato un valore non negativo. Unità: numero Statistiche valide: media, minimo, massimo, somma, esempi di dati (visualizzato come numero di esempi nella console di Amazon SQS) Non vi è alcun limite al numero di messaggi da inviare ai processi, tuttavia è possibile sottoporre questo backlog a un periodo di conservazione.

Metrica	Descrizione
`NumberOfEmptyReceives` [1]	Il numero di chiamate API `ReceiveMessage` che non hanno restituito un messaggio. Criteri di report: se la coda è attiva, viene riportato un valore non negativo. Unità: numero Statistiche valide: media, minimo, massimo, somma, esempi di dati (visualizzato come numero di esempi nella console di Amazon SQS)

Metrica	Descrizione
`NumberOfMessagesDeleted` [1]	Il numero dei messaggi eliminati dalla coda. Criteri di report: se la coda è attiva, viene riportato un valore non negativo. Unità: numero Statistiche valide: media, minimo, massimo, somma, esempi di dati (visualizzato come numero di esempi nella console di Amazon SQS) Amazon SQS genera il parametro `NumberOfMessagesDeleted` per ogni operazione di eliminazione riuscita che utilizza un handle di ricezione valido, incluse le eliminazioni duplicate. I seguenti scenari possono causare un valore del parametro `NumberOfMessagesDeleted` superiore al previsto: • Chiamando l'azione `DeleteMessage` su diversi handle di ricezione che appartengono allo stesso messaggio: se il messaggio non viene elaborato prima della scadenza del timeout visibilità, il messaggio diventa disponibile per altri consumatori che possono elaborarlo ed eliminarlo di nuovo, aumentando il valore del parametro `NumberOfMessagesDeleted`. • Chiamando l'azione `DeleteMessage` sullo stesso handle di ricezione

Metrica	Descrizione
	: se il messaggio viene elaborato ed eliminato ma chiami nuovamente l'azione `DeleteMessage` utilizzando lo stesso handle di ricezione, viene restituito uno stato riuscito, aumentando il valore del parametro `NumberOfMessagesDeleted` .
`NumberOfMessagesReceived` [1]	Il numero di messaggi restituiti da chiamate all'azione `ReceiveMessage` . Criteri di report: se la coda è attiva, viene riportato un valore non negativo. Unità: numero Statistiche valide: media, minimo, massimo, somma, esempi di dati (visualizzato come numero di esempi nella console di Amazon SQS)
`NumberOfMessagesSent` [1]	Il numero di messaggi aggiunti a una coda. Criteri di report: se la coda è attiva, viene riportato un valore non negativo. Unità: numero Statistiche valide: media, minimo, massimo, somma, esempi di dati (visualizzato come numero di esempi nella console di Amazon SQS)

Metrica	Descrizione
`SentMessageSize` [1]	La dimensione dei messaggi aggiunti a una coda. Criteri di report: se la coda è attiva, viene riportato un valore non negativo. Unità: byte Statistiche valide: media, minimo, massimo, somma, esempi di dati (visualizzato come numero di esempi nella console di Amazon SQS) **Note** `SentMessageSize` non viene visualizzato come metrica disponibile nella CloudWatch console finché non viene inviato almeno un messaggio alla coda corrispondente.

[1] Queste metriche sono calcolate dal punto di vista del servizio e possono includere nuovi tentativi. Non fare affidamento sui valori assoluti di queste metriche e non utilizzarle per stimare lo stato attuale della coda.

Dimensioni per le metriche Amazon SQS

L'unica dimensione a cui Amazon SQS invia è. CloudWatch `QueueName` Ciò significa che tutte le statistiche disponibili vengono filtrate per `QueueName`.

Convalida della conformità per Amazon SQS

Per sapere se il Servizio AWSè coperto da programmi di conformità specifici, consulta i Servizi AWScoperti dal programma di conformità e scegli il programma di conformità desiderato. Per informazioni generali, consulta Programmi per la conformità di AWS.

È possibile scaricare i report di audit di terze parti utilizzando AWS Artifact. Per ulteriori informazioni, consulta Download di report in AWS Artifact.

La responsabilità di conformità durante l'utilizzo dei Servizi AWSè determinata dalla riservatezza dei dati, dagli obiettivi di conformità dell'azienda e dalle normative vigenti. Per semplificare il rispetto della conformità, AWSmette a disposizione le seguenti risorse:

- Guide Quick Start per la sicurezza e conformità: queste guide all'implementazione illustrano considerazioni relative all'architettura e forniscono la procedura per l'implementazione di ambienti di base su AWSincentrati sulla sicurezza e sulla conformità.
- Architetture per la sicurezza e la conformità HIPAA su Amazon Web Services: questo whitepaper descrive come le aziende possono utilizzare AWSper creare applicazioni conformi alla normativa HIPAA.

  > **Note**
  >
  > Non tutti i Servizi AWSsono conformi ai requisiti HIPAA. Per ulteriori informazioni, consulta la sezione Riferimenti sui servizi conformi ai requisiti HIPAA.

- Risorse per la conformità AWS: una raccolta di cartelle di lavoro e guide suddivise per settore e area geografica.
- AWSGuide alla conformità dei clienti: comprendi il modello di responsabilità condivisa attraverso la lente della conformità. Le guide riassumono le migliori pratiche per la protezione Servizi AWS e mappano le linee guida per i controlli di sicurezza su più framework (tra cui il National Institute of Standards and Technology (NIST), il Payment Card Industry Security Standards Council (PCI) e l'International Organization for Standardization (ISO)).
- Valutazione delle risorse con le regole nella Guida per gli sviluppatori di AWS Config: il servizio AWS Configvaluta il livello di conformità delle configurazioni delle risorse con pratiche interne, linee guida e regolamenti.
- AWS Security Hub: questo Servizio AWSfornisce una visione completa dello stato di sicurezza all'interno di AWS. La Centrale di sicurezza utilizza i controlli di sicurezza per valutare le risorse

AWSe verificare la conformità agli standard e alle best practice del settore della sicurezza. Per un elenco dei servizi e dei controlli supportati, consulta la pagina Documentazione di riferimento sui controlli della Centrale di sicurezza.

- AWS Audit Manager: questo Servizio AWSaiuta a verificare continuamente l'utilizzo di AWSper semplificare la gestione dei rischi e della conformità alle normative e agli standard di settore.

Resilienza in Amazon SQS

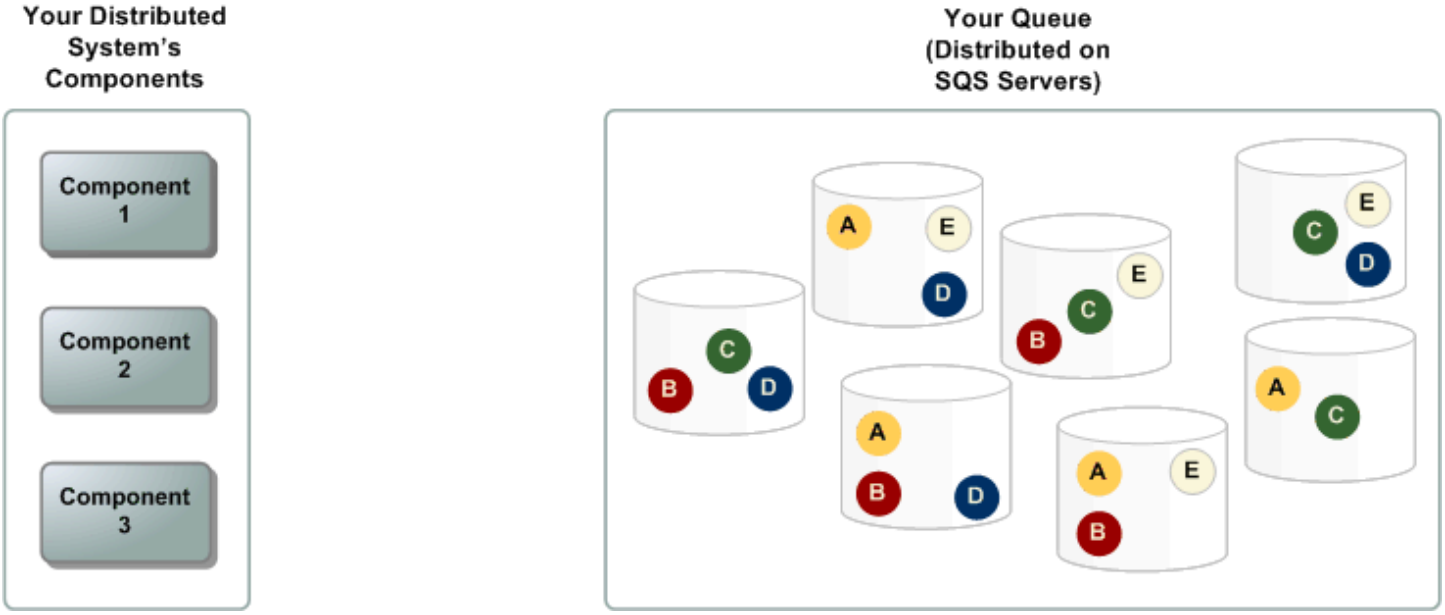
L'infrastruttura globale di AWS è basata su Regioni e zone di disponibilità AWS. AWS Le Regioni forniscono più zone di disponibilità fisicamente separate e isolate che sono connesse tramite reti altamente ridondanti, a bassa latenza e velocità di trasmissione effettiva elevata. Con le Zone di disponibilità, è possibile progettare e gestire applicazioni e database che eseguono il failover automatico tra zone di disponibilità senza interruzioni. Le Zone di disponibilità sono più disponibili, tolleranti ai guasti e scalabili, rispetto alle infrastrutture a data center singolo o multiplo. Per ulteriori informazioni sulle Regioni AWS e sulle zone di disponibilit, consulta Infrastruttura globale di AWS.

Oltre all'infrastruttura globale AWS, Amazon SQS offre code distribuite.

Code distribuite

Esistono tre parti principali in un sistema di messaggistica distribuito: i componenti del sistema distribuito, la coda (distribuita su server Amazon SQS) e i messaggi nella coda.

Nel seguente scenario, il sistema dispone di diversi produttori, ossia componenti che inviano messaggi alla coda, e consumatori, componenti che ricevono messaggi dalla coda. La coda (che contiene i messaggi da A a E) archivia in modo ridondante i messaggi su più server Amazon SQS.



Sicurezza dell'infrastruttura in Amazon SQS

In qualità di servizio gestito, Amazon SQS è protetto dalle procedure di sicurezza di rete globali AWS descritte nel whitepaper Amazon Web Services: Overview of Security Processes.

Utilizza le operazioni API pubblicate di AWS per accedere a Amazon SQS mediante la rete. I client devono supportare Transport Layer Security (TLS) 1.2 o versioni successive. I client devono, inoltre, supportare le suite di cifratura con PFS (Perfect Forward Secrecy), ad esempio Ephemeral Diffie-Hellman (DHE) o Elliptic Curve Ephemeral Diffie-Hellman (ECDHE).

Le richieste devono essere firmate utilizzando un ID chiave di accesso e una chiave di accesso segreta associate a un principal IAM. In alternativa, è possibile utilizzare AWS Security Token Service (AWS STS) per generare credenziali di sicurezza temporanee per le richieste di firma.

Puoi richiamare queste operazioni API da qualsiasi posizione di rete, ma Amazon SQS supporta le policy di accesso basate sulle risorse, che possono includere limitazioni in base all'indirizzo IP di origine. È inoltre possibile utilizzare le policy di Amazon SQS per controllare l'accesso da endpoint Amazon VPC o VPC specifici. Questo isola efficacemente l'accesso di rete a una coda Amazon SQS specifica solo dal VPC specifico all'interno della rete AWS. Per ulteriori informazioni, consulta Esempio 5: Negare l'accesso se non è un endpoint VPC.

Best practice di sicurezza per Amazon SQS

AWS fornisce molte caratteristiche di sicurezza per Amazon SQS, che è necessario esaminare nel contesto delle proprie policy di sicurezza.

> **Note**
>
> Le linee guida specifiche per l'implementazione fornite sono per i casi d'uso e le implementazioni comuni. Si consiglia di visualizzare best practice nel contesto del caso d'uso specifico, dell'architettura e del modello di minaccia.

Best practice di prevenzione

Di seguito sono riportate le best practice di prevenzione per la sicurezza per Amazon SQS.

Argomenti

Assicurarsi che le code non siano accessibili pubblicamente

A meno che tu non richieda esplicitamente a chiunque su Internet di essere in grado di leggere o scrivere nella coda Amazon SQS, dovresti assicurarti che la coda non sia accessibile pubblicamente (accessibile da tutti nel mondo o da qualsiasi utente AWS autenticato).

- Evitare di creare policy con `Principal` impostato su `""`.
- Evitare di utilizzare un carattere jolly (`*`). Assegnare invece un nome a uno o più utenti specifici.

Implementazione dell'accesso con privilegi minimi

Quando si concedono autorizzazioni, si decide chi le riceve, per quali code sono le autorizzazioni e operazioni API specifiche che si desidera consentire per queste code. Implementare privilegi minimi è importante per ridurre i rischi per la sicurezza e ridurre l'effetto di errori o intenti dannosi.

Seguire i consigli di sicurezza standard relativi alla concessione dei privilegi minimi. Cioè, concedere solo le autorizzazioni necessarie per eseguire un'attività specifica. È possibile effettuare questa implementazione utilizzando una combinazione di policy di sicurezza.

Amazon SQS utilizza il modello produttore-consumatore, che richiede tre tipi di accesso all'account utente:

- Amministratori: accesso alla creazione, alla modifica e all'eliminazione di code. Gli amministratori controllano anche le policy della coda.
- Produttori: accesso all'invio di messaggi alle code.
- Consumatori: accesso alla ricezione e all'eliminazione di messaggi dalle code.

Per ulteriori informazioni, consulta le sezioni seguenti:

- Identity and Access Management in Amazon SQS
- Autorizzazioni API Amazon SQS: riferimento a operazioni e risorse
- Utilizzo di policy personalizzate con la sintassi delle policy di accesso Amazon SQS

Utilizzare ruoli IAM per le applicazioni e i servizi AWS che richiedono l'accesso Amazon SQS

Per applicazioni o servizi AWS, ad esempio Amazon EC2, per l'accesso alle code Amazon SQS, è necessario utilizzare credenziali AWS valide nelle richieste API AWS. Poiché queste credenziali non vengono ruotate automaticamente, non è necessario archiviare le credenziali AWS direttamente nell'applicazione o nell'istanza EC2.

È preferibile usare un ruolo IAM per gestire credenziali provvisorie per le applicazioni o i servizi che devono accedere ad Amazon SQS. Quando utilizzi un ruolo, non devi necessariamente distribuire credenziali a lungo termine (come, ad esempio, nome utente e password e chiavi di accesso) a un'istanza EC2, o un servizio AWS come AWS Lambda. Al contrario, il ruolo fornisce autorizzazioni provvisorie che possono essere utilizzate dalle applicazioni durante le chiamate ad altre risorse AWS.

Per ulteriori informazioni, consultaRuoli IAM e Scenari comuni per ruoli: utenti, applicazioni e servizi nella Guida per l'utente IAM.

Implementare la crittografia lato server

Per attenuare i problemi di perdita di dati, utilizzare la crittografia dei dati inattivi per crittografare i messaggi utilizzando una chiave memorizzata in un percorso diverso da quello in cui vengono archiviati i messaggi. La crittografia lato server (SSE) fornisce la crittografia dei dati inattivi. Amazon SQS esegue la crittografia dei dati a livello di messaggio quando li memorizza e decrittografa i messaggi per l'utente quando vi accede. SSE utilizza le chiavi gestite in AWS Key Management Service. Finché si autentica la richiesta e si dispone delle autorizzazioni di accesso, non vi è alcuna differenza tra l'accesso alle code crittografate e non crittografate.

Per ulteriori informazioni, consulta Crittografia a riposo e Gestione delle chiavi.

Applica la crittografia dei dati in transito

Senza HTTPS (TLS), un utente malintenzionato basato sulla rete può spiare il traffico di rete o manipolarlo, utilizzando un attacco di tipo man-in-the-middle. Consenti solo connessioni crittografate tramite HTTPS (TLS) utilizzando la condizione`aws:SecureTransport` nella policy della coda per forzare le richieste a utilizzare SSL.

Prendere in considerazione l'utilizzo di endpoint VPC per accedere a Amazon SQS

Se si dispone di code con cui è necessario essere in grado di interagire ma che non devono assolutamente essere esposte a Internet, utilizzare gli endpoint VPC per accodare l'accesso solo agli host all'interno di un determinato VPC. È possibile utilizzare le policy della coda per controllare l'accesso alle code da endpoint Amazon VPC specifici o da VPC specifici.

Gli endpoint VPC di Amazon SQS offrono due modi per controllare l'accesso ai messaggi:

- È possibile controllare le richieste, gli utenti o i gruppi autorizzati tramite un endpoint VPC specifico.
- È possibile controllare quali VPC o endpoint VPC hanno accesso alla coda utilizzando una policy della coda.

Per ulteriori informazioni, consulta Endpoint di Amazon Virtual Private Cloud per Amazon SQS e Creazione di una policy per endpoint VPC di Amazon per Amazon SQS.

Lavorare con le API Amazon SQS

Questa sezione fornisce informazioni sulla costruzione degli endpoint Amazon SQS, sull'esecuzione di richieste delle API di query utilizzando i metodi GET e POST e sull'utilizzo di operazioni API in batch. Per informazioni dettagliate sulle azioni di Amazon SQS, inclusi parametri, errori, esempi e tipi di dati, consulta Guida di riferimento delle API Amazon Simple Queue Service.

Per accedere ad Amazon SQS utilizzando un'ampia gamma di linguaggi di programmazione, puoi anche usare gli SDK AWS che contengono la seguente funzionalità automatica:

- Firma crittografica delle richieste di servizio
- Nuovi tentativi di richiesta
- Gestione delle risposte di errore

Per informazioni sullo strumento a riga di comando, consulta le sezioni Amazon SQS nella Guida di riferimento AWS CLI e lao AWS Tools for PowerShellGuida di riferimento Cmlet.

API Amazon SQS con protocollo JSON AWS

Amazon SQS utilizza il protocollo AWS JSON come meccanismo di trasporto per tutte le API di Amazon SQS nelle AWSversioni SDK specificate. AWS Il protocollo JSON offre un throughput più elevato, una latenza inferiore e una comunicazione più rapida. application-to-application AWS Il protocollo JSON è più efficiente nella serializzazione/deserializzazione di richieste e risposte rispetto al protocollo di query AWS. Se preferisci comunque utilizzare il protocollo di query AWS con le API SQS, consulta Quali linguaggi sono supportati per il protocollo AWS JSON utilizzato nelle API di Amazon SQS? per le versioni AWS SDK che supportano il protocollo di query Amazon SQS AWS.

Amazon SQS utilizza il protocollo AWS JSON per comunicare tra i client AWS SDK (ad esempio, Java, Python, Golang) e JavaScript il server Amazon SQS. Una richiesta HTTP di un'operazione API Amazon SQS accetta input in formato JSON. L'operazione Amazon SQS viene eseguita e la risposta di esecuzione viene inviata al client SDK in formato JSON. Rispetto alle query AWS, AWS JSON è più semplice, veloce ed efficiente per il trasporto dei dati tra client e server.

- Il protocollo JSON AWS funge da mediatore tra il client e il server Amazon SQS.
- Il server non comprende il linguaggio di programmazione in cui viene creata l'operazione Amazon SQS, ma comprende il protocollo AWS JSON.

- Il protocollo AWS JSON utilizza la serializzazione (conversione dell'oggetto in formato JSON) e la deserializzazione (conversione del formato JSON in oggetto) tra client e server Amazon SQS.

Per ulteriori informazioni sul protocollo AWS JSON con Amazon SQS, consulta Domande frequenti sul protocollo AWS JSON di Amazon SQS.

Il protocollo AWS JSON è disponibile nella versione SDK AWS specificata. Per esaminare la versione e le date di rilascio dell'SDK nelle diverse varianti linguistiche, consulta la matrice di supporto delle versioni degli AWS SDK e degli strumenti nella Guida di riferimento agli strumenti e agli AWS SDK

Argomenti

Effettuare richieste API di query utilizzando il protocollo AWS JSON

In questa sezione apprenderai come creare un endpoint Amazon SQS, eseguire le richieste POST e interpretare le risposte.

> **Note**
>
> Il protocollo AWS JSON è supportato per la maggior parte delle varianti linguistiche. Per un elenco delle varianti di linguaggi supportate, consulta Quali linguaggi sono supportati per il protocollo AWS JSON utilizzato nelle API di Amazon SQS?.

Argomenti

Costruzione di un endpoint

Per utilizzare le code Amazon SQS, è necessario creare un endpoint. Per informazioni sugli endpoint Amazon SQS, consulta le seguenti pagine in Riferimenti generali di Amazon Web Services:

- Endpoint regionali
- Endpoint e quote di Amazon Simple Queue Service

Ogni endpoint Amazon SQS è completamente indipendente. Ad esempio, se due code sono denominate MyQueue e una ha l'endpoint `sqs.us-east-2.amazonaws.com`, mentre l'altra ha l'endpoint `sqs.eu-west-2.amazonaws.com`, le due code non condividono alcun dato tra loro.

Di seguito è riportato un esempio di un endpoint che invia una richiesta per creare una coda.

```
POST / HTTP/1.1
Host: sqs.us-west-2.amazonaws.com
X-Amz-Target: AmazonSQS.CreateQueue
X-Amz-Date: <Date>
Content-Type: application/x-amz-json-1.0
Authorization: <AuthParams>
Content-Length: <PayloadSizeBytes>
Connection: Keep-Alive
{
    "QueueName":"MyQueue",
    "Attributes": {
        "VisibilityTimeout": "40"
    },
    "tags": {
        "QueueType": "Production"
    }
}
```

> **Note**
>
> Per i nomi e gli URL delle code viene fatta la distinzione tra maiuscole e minuscole.
> La struttura di *AUTHPARAMS* dipende dalla modalità di firma della richiesta API. Per ulteriori informazioni, consulta Firma delle richieste API AWS in Riferimento generale per Amazon Web Services.

Effettuare una richiesta POST

Una richiesta Amazon SQS POST invia i parametri di query come modulo nel corpo di una richiesta HTTP.

Di seguito è riportato un esempio di intestazione HTTP con `X-Amz-Target` impostato su `AmazonSQS.<operationName>` e di intestazione HTTP con `Content-Type` impostato su `application/x-amz-json-1.0`.

```
POST / HTTP/1.1
Host: sqs.<region>.<domain>
X-Amz-Target: AmazonSQS.SendMessage
X-Amz-Date: <Date>
Content-Type: application/x-amz-json-1.0
Authorization: <AuthParams>
Content-Length: <PayloadSizeBytes>
Connection: Keep-Alive
{
    "QueueUrl": "https://sqs.<region>.<domain>/<awsAccountId>/<queueName>/",
    "MessageBody": "This is a test message",
}
```

Questa richiesta HTTP POST invia un messaggio a una coda Amazon SQS.

> **Note**
>
> Entrambe le intestazioni HTTP `X-Amz-Target` e `Content-Type` sono obbligatorie. Il client HTTP potrebbe aggiungere altri elementi alla richiesta HTTP, a seconda della versione HTTP del client.

Interpretazione delle risposte dell'API JSON di Amazon SQS

Quando risponde a una richiesta di azione, Amazon SQS restituisce una struttura di dati JSON che contiene i risultati della richiesta. Per ulteriori informazioni, consulta le azioni singole nella Documentazione di riferimento delle API di Amazon Simple Queue Service e Domande frequenti sul protocollo AWS JSON di Amazon SQS.

Argomenti

- Struttura di una risposta JSON corretta

Struttura di una risposta JSON corretta

Se la richiesta ha esito positivo, l'elemento di risposta principale è `x-amzn-RequestId`, che contiene l'Universal Unique Identifier (UUID) della richiesta e altri campi di risposta aggiunti. Ad esempio, l'elemento `CreateQueue` contiene il campo `QueueUrl` che, a sua volta, contiene l'URL della coda creata.

```
HTTP/1.1 200 OK
x-amzn-RequestId: <requestId>
Content-Length: <PayloadSizeBytes>
Date: <Date>
Content-Type: application/x-amz-json-1.0
{
    "QueueUrl":"https://sqs.us-east-1.amazonaws.com/111122223333/MyQueue"
}
```

Struttura di una risposta di errore JSON

Se una richiesta genera un errore, Amazon SQS restituisce la risposta principale, inclusi l'intestazione HTTP e il corpo.

Nell'intestazione HTTP, `x-amzn-RequestId` contiene l'UUID della richiesta. `x-amzn-query-error` contiene due informazioni: il tipo di errore e se si tratta di un errore del produttore o del consumatore.

Nel corpo della risposta, `"__type"` indica altri dettagli sull'errore e `Message` indica la condizione di errore in un formato leggibile.

Di seguito è riportato un esempio di risposta di errore in formato JSON:

```
HTTP/1.1 400 Bad Request
x-amzn-RequestId: 66916324-67ca-54bb-a410-3f567a7a0571
x-amzn-query-error: AWS.SimpleQueueService.NonExistentQueue;Sender
Content-Length: <PayloadSizeBytes>
Date: <Date>
Content-Type: application/x-amz-json-1.0
{
    "__type": "com.amazonaws.sqs#QueueDoesNotExist",
    "message": "The specified queue does not exist."
```

```
}
```

Domande frequenti sul protocollo AWS JSON di Amazon SQS

Domande frequenti sull'uso del protocollo AWS JSON con Amazon SQS.

Cos'è il protocollo AWS JSON e in che cosa differisce dalle richieste e dalle risposte API Amazon SQS esistenti?

JSON è uno dei metodi di cablaggio più utilizzati e accettati per la comunicazione tra sistemi eterogenei. Amazon SQS utilizza JSON come mezzo per comunicare tra un client AWS SDK (ad esempio, Java, Python, Golang) JavaScript e il server Amazon SQS. Una richiesta HTTP di un'operazione API Amazon SQS accetta input in formato JSON. L'operazione Amazon SQS viene eseguita e la risposta di esecuzione viene inviata al client SDK in formato JSON. Rispetto alle query AWS, JSON è più efficiente in termini di trasporto dei dati tra client e server.

- Il protocollo AWS JSON di Amazon SQS opera da mediatore tra il client e il server Amazon SQS.
- Il server non comprende il linguaggio di programmazione in cui viene creata l'operazione Amazon SQS, ma comprende il protocollo AWS JSON.
- Il protocollo AWS JSON di Amazon SQS utilizza la serializzazione (conversione dell'oggetto in formato JSON) e la deserializzazione (conversione del formato JSON in oggetto) tra client e server Amazon SQS.

Come posso iniziare a usare i protocolli AWS JSON per Amazon SQS?

Per iniziare a usare l'ultima versione di AWS SDK e ottenere una messaggistica più veloce per Amazon SQS, aggiorna AWS SDK alla versione specificata o a qualsiasi versione successiva. Per ulteriori informazioni sui client SDK, consulta la colonna Guida nella tabella seguente.

Di seguito è riportato un elenco di versioni SDK in diverse varianti linguistiche per il protocollo AWS JSON da utilizzare con le API di Amazon SQS:

Lingua	Archivio client SDK	Versione del client SDK richiesta	Guida
C++	aws/ aws-sdk-cpp	1.11,98	SDK AWS per C++

Lingua	Archivio client SDK	Versione del client SDK richiesta	Guida
Golang 1.x	seghe/aws-sdk-go	v1.47.7	SDK AWS per Go
Golang 2.x	sega/ 2 aws-sdk-go-v	v1.28.0	AWS SDK per Go V2
Java 1.x	sega/ aws-sdk-java	1,12.585	SDK AWS per Java
Java 2.x	seghe/2 aws-sdk-j ava-v	2,21,19	SDK AWS per Java
JavaScript v2.x	aws/ aws-sdk-js	v2.1492.0	JavaScript su AWS
JavaScript v3.x	aws/ 3 aws-sdk-js-v	v3.447.0	JavaScript su AWS
.NET	seghe/aws-sdk-net	3,7681,0	SDK AWS per .NET
PHP	seghe/aws-sdk-php	3,285,2	SDK AWS per PHP
Python-boto3	boto/boto3	1,28,82	AWS SDK per Python (Boto3)
Python-botocore	boto/botocore	1,31,82	AWS SDK per Python (Boto3)
awscli	CLI AWS	1,29,82	Interfaccia a riga di comando AWS
Ruby	seghe/aws-sdk-ruby	1.67,0	SDK AWS per Ruby

Quali sono i rischi dell'abilitazione del protocollo JSON per i miei carichi di lavoro Amazon SQS?

Se utilizzi un'implementazione personalizzata di AWS SDK o una combinazione di client personalizzati e AWS SDK per interagire con Amazon SQS che genera risposte basate su Query AWS (ovvero basate su XML), l'implementazione potrebbe essere incompatibile con il protocollo JSON AWS. Se riscontri problemi, contatta l'assistenza AWS.

Cosa succede se uso già la versione AWS SDK più recente, ma la mia soluzione open source non supporta JSON?

È necessario modificare la versione dell'SDK con la versione precedente a quella in uso. Per ulteriori informazioni, consulta Come posso iniziare a usare i protocolli AWS JSON per Amazon SQS?. Le versioni di AWS SDK elencate in Come posso iniziare a usare i protocolli AWS JSON per Amazon SQS? utilizzano il protocollo JSON wire per le API di Amazon SQS. Se modifichi AWS SDK alla versione precedente, le tue API Amazon SQS utilizzeranno la query AWS.

Quali linguaggi sono supportati per il protocollo AWS JSON utilizzato nelle API di Amazon SQS?

Amazon SQS supporta tutte le varianti di linguaggio in cui gli AWS SDK sono generalmente disponibili (GA). Al momento non supportiamo Kotlin, Rust o Swift. Per saperne di più sulle altre varianti linguistiche, consulta Strumenti per costruire su AWS.

Quali regioni sono supportate per il protocollo AWS JSON utilizzato nelle API di Amazon SQS?

Amazon SQS supporta il protocollo AWS JSON in tutte le regioni AWS in cui è disponibile Amazon SQS.

Quali miglioramenti della latenza posso aspettarmi eseguendo l'aggiornamento alle versioni AWS SDK specificate per Amazon SQS utilizzando il protocollo AWS JSON?

Il protocollo AWS JSON è più efficiente in termini di serializzazione/deserializzazione delle richieste e delle risposte rispetto al protocollo di query AWS. In base ai test AWS delle prestazioni per un payload di messaggi di 5 KB, il protocollo JSON per Amazon SQS end-to-end riduce la latenza di elaborazione dei messaggi fino al 23% e riduce l'utilizzo della CPU e della memoria lato client dell'applicazione.

Il protocollo di query AWS diventerà obsoleto?

Il protocollo di query AWS continuerà a essere supportato. È possibile continuare a utilizzare il protocollo di query AWS purché la versione di AWS SDK sia impostata su una versione precedente diversa da quella elencata in Come posso iniziare a usare i protocolli AWS JSON per Amazon SQS?.

Dove posso trovare ulteriori informazioni sul protocollo AWS JSON?

Puoi trovare ulteriori informazioni sul protocollo JSON nel AWS protocollo JSON 1.0 nella documentazione Smithy. Per ulteriori informazioni sulle richieste API di Amazon SQS che utilizzano il protocollo AWS JSON, consulta Effettuare richieste API di query utilizzando il protocollo AWS JSON.

Effettuare richieste API Query con il protocollo di query AWS

In questa sezione apprenderai come creare un endpoint Amazon SQS, eseguire le richieste POST e GET e interpretare le risposte.

Argomenti

Costruzione di un endpoint

Per lavorare con le code Amazon SQS, è necessario creare un endpoint. Per informazioni sugli endpoint Amazon SQS, consulta le seguenti pagine in Riferimenti generali di Amazon Web Services:

- Endpoint regionali
- Endpoint e quote di Amazon Simple Queue Service

Ogni endpoint Amazon SQS è completamente indipendente. Ad esempio, se due code sono denominate MyQueue e una ha l'endpoint `sqs.us-east-2.amazonaws.com` mentre l'altra ha l'endpoint `sqs.eu-west-2.amazonaws.com`, le due code non condividono alcun dato tra loro.

Di seguito è riportato un esempio di un endpoint che invia una richiesta per creare una coda.

```
https://sqs.eu-west-2.amazonaws.com/
```

```
?Action=CreateQueue
&DefaultVisibilityTimeout=40
&QueueName=MyQueue
&Version=2012-11-05
&AUTHPARAMS
```

> **Note**
>
> Per i nomi e gli URL delle code viene fatta la distinzione tra maiuscole e minuscole.
> La struttura di *AUTHPARAMS* dipende dalla modalità di firma della richiesta API. Per ulteriori informazioni, consulta Firma delle richieste API AWS in Riferimento generale per Amazon Web Services.

Effettuare una richiesta GET

Una richiesta Amazon SQS GET è strutturata come un URL che contiene le seguenti informazioni:

- Endpoint: la risorsa su cui agisce la richiesta (il nome della coda e l'URL), ad esempio: `https://sqs.us-east-2.amazonaws.com/123456789012/MyQueue`
- Operazione: l'operazione che desideri eseguire sull'endpoint. Un punto di domanda (?) separa l'endpoint dall'azione, ad esempio: `?Action=SendMessage&MessageBody=Your%20Message%20Text`
- Parametri: gli eventuali parametri della richiesta. Ogni parametro è separato da una e commerciale (&) ad esempio: `&Version=2012-11-05&AUTHPARAMS`

Il seguente è un esempio di una richiesta GET per inviare un messaggio a una coda Amazon SQS.

```
https://sqs.us-east-2.amazonaws.com/123456789012/MyQueue
?Action=SendMessage&MessageBody=Your%20message%20text
&Version=2012-11-05
&AUTHPARAMS
```

> **Note**
>
> Per i nomi e gli URL delle code viene fatta la distinzione tra maiuscole e minuscole.
> Poiché le richieste GET sono URL, i valori di tutti i parametri devono essere codificati in formato URL. Considerato che nell'URL non sono consentiti spazi, ogni spazio è codificato

nell'URL come `%20`. Il resto dell'esempio non è stato codificato nell'URL per renderne più facile la lettura.

Effettuare una richiesta POST

Una richiesta Amazon SQS POST invia i parametri di query come modulo nel corpo di una richiesta HTTP.

Di seguito è riportato un esempio di intestazione HTTP con `Content-Type` impostato su `application/x-www-form-urlencoded`.

```
POST /123456789012/MyQueue HTTP/1.1
Host: sqs.us-east-2.amazonaws.com
Content-Type: application/x-www-form-urlencoded
```

L'intestazione è seguita da una richiesta GET `form-urlencoded` che invia un messaggio a una coda Amazon SQS. Ogni parametro è separato da una e commerciale (&).

```
Action=SendMessage
&MessageBody=Your+Message+Text
&Expires=2020-10-15T12%3A00%3A00Z
&Version=2012-11-05
&AUTHPARAMS
```

> **Note**
>
> Solo l'intestazione HTTP `Content-Type` è obbligatoria. La richiesta *AUTHPARAMS* è la stessa della richiesta GET.
> Il client HTTP potrebbe aggiungere altri elementi alla richiesta HTTP, a seconda della versione HTTP del client.

Interpretazione delle risposte dell'API XML di Amazon SQS

In risposta a una richiesta di azione, Amazon SQS restituisce una struttura di dati XML che contiene i risultati della richiesta. Per ulteriori informazioni, consulta le operazioni singole nella Documentazione di riferimento delle API di Amazon Simple Queue Service.

Argomenti

Struttura di una risposta XML corretta

Se la richiesta è andata a buon fine, l'elemento principale della risposta prende il nome dell'azione, con la dicitura `Response` aggiunta (ad esempio, *`ActionName`*`Response`).

Questo elemento contiene i seguenti elementi figli:

- ***`ActionName`*`Result`**: contiene un elemento specifico dell'operazione. Ad esempio, l'elemento `CreateQueueResult` contiene l'elemento `QueueUrl` che, a sua volta, contiene l'URL della coda creata.
- **`ResponseMetadata`**: contiene `RequestId` che, a sua volta, contiene l'UUID della richiesta.

Di seguito è riportato un esempio di risposta corretta in formato XML:

```
<CreateQueueResponse
   xmlns=https://sqs.us-east-2.amazonaws.com/doc/2012-11-05/
   xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
   xsi:type=CreateQueueResponse>
   <CreateQueueResult>
      <QueueUrl>https://sqs.us-east-2.amazonaws.com/770098461991/queue2</QueueUrl>
   </CreateQueueResult>
   <ResponseMetadata>
      <RequestId>cb919c0a-9bce-4afe-9b48-9bdf2412bb67</RequestId>
   </ResponseMetadata>
</CreateQueueResponse>
```

Struttura di una risposta di errore XML

Se una richiesta ha esito negativo, Amazon SQS restituisce sempre il principale elemento di risposta `ErrorResponse`. Questo elemento contiene un elemento `Error` e un elemento `RequestId`.

L'elemento `Error` contiene i seguenti elementi figli:

- **`Type`**: specifica se l'errore è stato un errore di produttore o consumatore.

- **`Code`**: specifica il tipo di errore.
- **`Message`**: specifica la condizione dell'errore in un formato leggibile.
- **`Detail`**: (Facoltativo) Specifica ulteriori dettagli sull'errore.

L'elemento `RequestId` contiene l'UUID della richiesta.

Di seguito è riportato un esempio di risposta di errore in formato XML:

```
<ErrorResponse>
   <Error>
      <Type>Sender</Type>
      <Code>InvalidParameterValue</Code>
      <Message>
         Value (quename_nonalpha) for parameter QueueName is invalid.
         Must be an alphanumeric String of 1 to 80 in length.
      </Message>
   </Error>
   <RequestId>42d59b56-7407-4c4a-be0f-4c88daeea257</RequestId>
</ErrorResponse>
```

Autenticazione di richieste

L'autenticazione è un processo per l'identificazione e la verifica di chi invia una richiesta. Durante la prima fase di autenticazione, AWS verifica l'identità del produttore e se il produttore è registrato per utilizzare AWS (per ulteriori informazioni, consulta Fase 1: creazione di un Account AWS e di un utente IAM). Quindi, AWS si attiene alla seguente procedura:

1. Il produttore (mittente) ottiene le credenziali necessarie.
2. Il produttore invia al consumatore (ricevitore) una richiesta con la credenziale.
3. Il consumatore utilizza la credenziale per verificare se il produttore ha inviato la richiesta.
4. Si verifica una delle seguenti situazioni:
   - Se l'autenticazione va a buon fine, il consumatore elabora la richiesta.
   - Se l'autenticazione ha esito negativo, il consumatore rifiuta la richiesta e restituisce un errore.

Argomenti

- Processo di autenticazione di base con HMAC-SHA

- Parte 1: la richiesta dall'utente
- Parte 2: la risposta di AWS

Processo di autenticazione di base con HMAC-SHA

Quando accedi ad Amazon SQS utilizzando l'API della query, devi fornire le seguenti voci affinché la richiesta possa essere autenticata:

- L'ID chiave di accesso AWS che identifica l'account Account AWS utilizzato da AWS per ricercare la chiave di accesso segreta.
- La firma della richiesta HMAC-SHA, calcolata utilizzando la tua chiave di accesso segreta (un segreto condiviso noto solo a te e a AWS; per ulteriori informazioni, consulta RFC2104). L'AWS SDK gestisce il processo di firma; tuttavia, se invii una richiesta di query su HTTP o HTTPS, devi includere una firma in ogni richiesta di query.

1. Ottenere una chiave di firma Signature Version 4. Per ulteriori informazioni, consulta Ottenere una chiave di firma con Java.

   > **Note**
   >
   > Amazon SQS supporta Signature Version 4, che offre una sicurezza basata su SHA256 migliorata e prestazioni più elevate rispetto alle versioni precedenti. Se crei nuove applicazioni che utilizzano Amazon SQS, è consigliabile usare Signature Version 4.

2. Le firme di richiesta devono essere codificate in base64. Il seguente codice Java di esempio procede in questo modo:

```
package amazon.webservices.common;

// Define common routines for encoding data in AWS requests.
public class Encoding {

    /* Perform base64 encoding of input bytes.
     * rawData is the array of bytes to be encoded.
     * return is the base64-encoded string representation of rawData.
     */
    public static String EncodeBase64(byte[] rawData) {
        return Base64.encodeBytes(rawData);
```

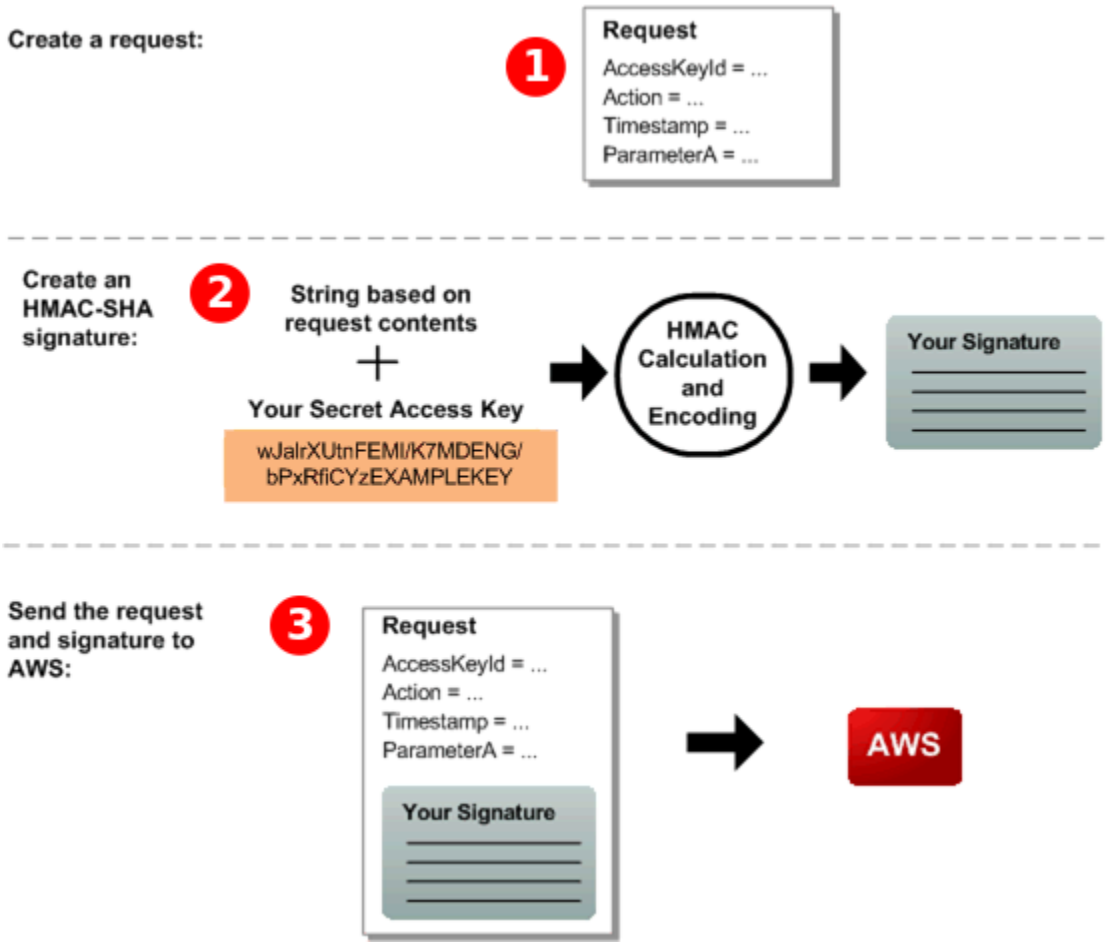
```
    }
}
```

- Il timestamp (o scadenza) della richiesta. Il timestamp utilizzato nella richiesta deve essere un `dateTime` oggetto, con la data completa, incluso ore, minuti e secondi. Ad esempio: `2007-01-31T23:59:59Z` Anche se non è obbligatorio, ti consigliamo di fornire l'oggetto nell'orario UTC (fuso orario di Greenwich).

> **Note**
>
> Assicurati che l'orario del tuo server sia impostato correttamente. Se specifichi un timestamp (anziché una scadenza), la richiesta scade automaticamente 15 minuti dopo l'orario specificato (AWS non elabora una richiesta se il timestamp è precedente all'ora corrente sui server AWS per più di 15 minuti).
> Se usi .NET, non devi inviare timestamp troppo specifici (poiché diverse interpretazioni della precisione del tempo aggiuntivo devono essere ignorate). In questo caso, è necessario costruire manualmente oggetti `dateTime` con una precisione non superiore a un millisecondo.

Parte 1: la richiesta dall'utente

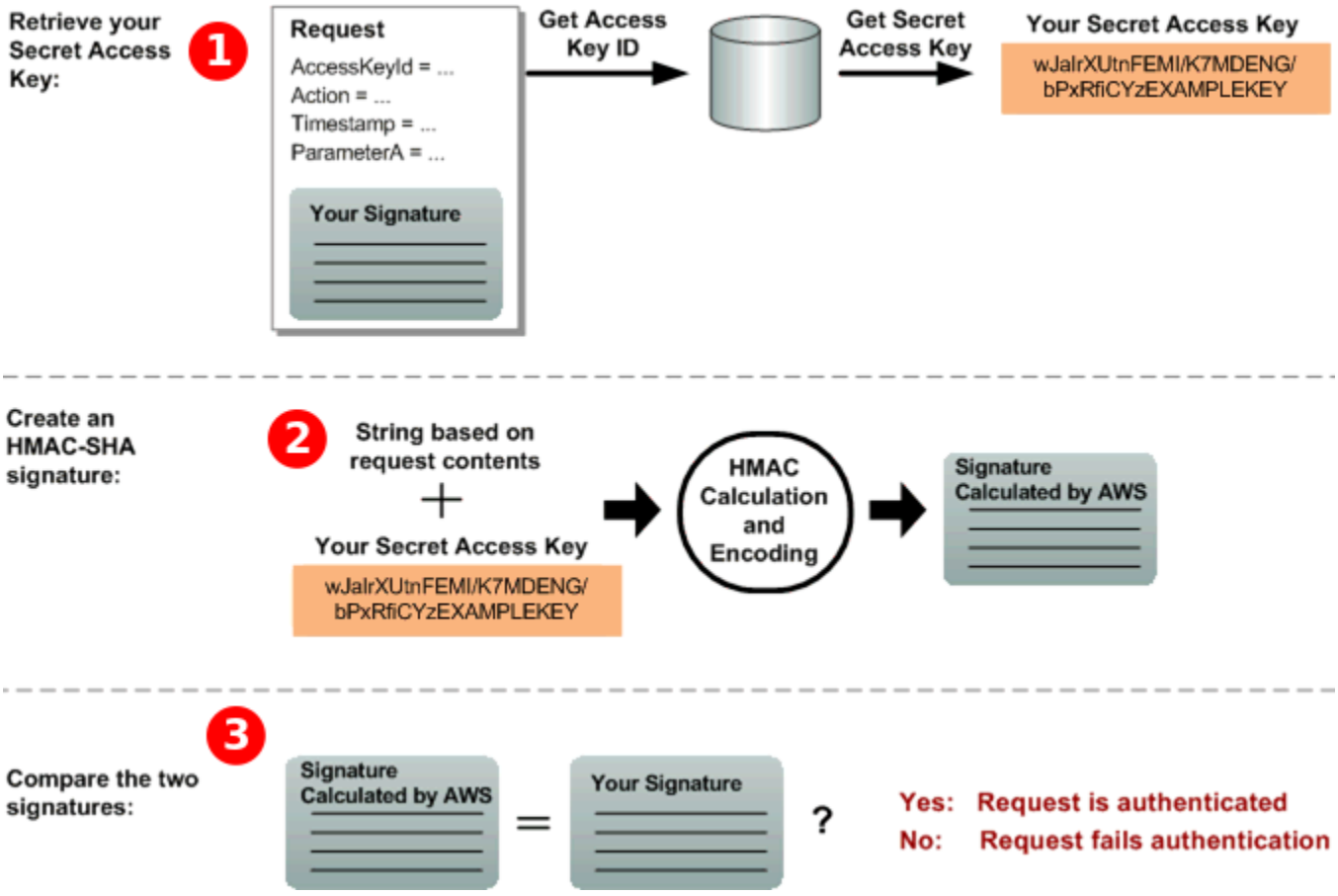
Di seguito viene illustrato il processo necessario per autenticare le richieste AWS utilizzando una firma della richiesta HMAC-SHA.



1. Costruisci una richiesta da inoltrare ad AWS.
2. Calcola una firma (HMAC-SHA) con il codice di autenticazione del messaggio di hash con chiave utilizzando la tua Secret Access Key.
3. Includi la firma e l'ID chiave di accesso nella richiesta, quindi invia la richiesta ad AWS.

Parte 2: la risposta di AWS

AWS inizia il seguente processo in risposta.



1. AWS utilizza l'ID chiave di accesso per cercare la chiave di accesso segreta.
2. AWS genera una firma a partire dai dati della richiesta e dalla chiave di accesso segreta utilizzando lo stesso algoritmo utilizzato per calcolare la firma inviata nella richiesta.
3. Si verifica una delle seguenti situazioni:
   - Se la firma generata da AWS corrisponde a quella inviata nella richiesta, la richiesta viene considerata da AWS come autentica.
   - Se il confronto non va a buon fine, la richiesta viene scartata e AWS restituisce un errore.

Operazioni in batch per Amazon SQS

Per ridurre i costi o modificare fino a 10 messaggi con una sola operazione, puoi utilizzare le seguenti operazioni in batch:

- `SendMessageBatch`
- `DeleteMessageBatch`

- `ChangeMessageVisibilityBatch`

Puoi sfruttare la funzionalità batch utilizzando l'API della query o un SDK AWS che supporta le operazioni in batch di Amazon SQS.

> **Note**
>
> La dimensione totale di tutti i messaggi inviati in una singola chiamata `SendMessageBatch` non può superare 262.144 byte (256 KB).
> Non è possibile impostare le autorizzazioni per `SendMessageBatch`, `DeleteMessageBatch`oppure `ChangeMessageVisibilityBatch` esplicitamente. Impostare le autorizzazioni per `SendMessage`, `DeleteMessage` o `ChangeMessageVisibility` consente di impostare le autorizzazioni per le versioni in batch corrispondenti di tali azioni.
> La console Amazon SQS non supporta le azioni in batch.

Argomenti

Attivazione del buffering lato client e del batching di richieste

AWS SDK for Java include `AmazonSQSBufferedAsyncClient` che accede ad Amazon SQS. Questo client facilita il batching semplice delle richieste abilitando il buffering lato client, dove le chiamate effettuate dal client vengono prima memorizzate, quindi inviate come una richiesta di batch ad Amazon SQS.

Il buffering lato client consente di memorizzare e inviare fino a 10 richieste in batch, riducendo il costo di utilizzo di Amazon SQS e il numero di richieste inviate. `AmazonSQSBufferedAsyncClient` effettua il buffering delle chiamate sincrone e asincrone. Le richieste in batch e il supporto per il polling lungo possono inoltre contribuire ad aumentare il throughput. Per ulteriori informazioni, consulta Aumentare il throughput con il dimensionamento orizzontale e il raggruppamento delle operazioni.

Poiché `AmazonSQSBufferedAsyncClient` implementa la stessa interfaccia di `AmazonSQSAsyncClient`, la migrazione da `AmazonSQSAsyncClient` a `AmazonSQSBufferedAsyncClient` di solito richiede solo modifiche minime al codice esistente.

> **Note**
>
> L'Amazon SQS Buffered Asynchronous Client attualmente non supporta le code FIFO.

Argomenti

Utilizzo di AmazonSQSBufferedAsyncClient

Prima di iniziare, completa i passaggi descritti in Configurazione di Amazon SQS.

> **Important**
>
> AWS SDK for Java 2.x non è al momento compatibile con `AmazonSQSBufferedAsyncClient`.

Puoi creare un nuovo `AmazonSQSBufferedAsyncClient` basato su `AmazonSQSAsyncClient`, ad esempio:

```
// Create the basic Amazon SQS async client
final AmazonSQSAsync sqsAsync = new AmazonSQSAsyncClient();

// Create the buffered client
final AmazonSQSAsync bufferedSqs = new AmazonSQSBufferedAsyncClient(sqsAsync);
```

Una volta creato il nuovo `AmazonSQSBufferedAsyncClient`, è possibile utilizzarlo per l'invio di più richieste ad Amazon SQS; (proprio come con `AmazonSQSAsyncClient`), ad esempio:

```
final CreateQueueRequest createRequest = new
 CreateQueueRequest().withQueueName("MyQueue");
```

```
final CreateQueueResult res = bufferedSqs.createQueue(createRequest);

final SendMessageRequest request = new SendMessageRequest();
final String body = "Your message text" + System.currentTimeMillis();
request.setMessageBody( body );
request.setQueueUrl(res.getQueueUrl());

final Future<SendMessageResult> sendResult = bufferedSqs.sendMessageAsync(request);

final ReceiveMessageRequest receiveRq = new ReceiveMessageRequest()
    .withMaxNumberOfMessages(1)
    .withQueueUrl(queueUrl);
final ReceiveMessageResult rx = bufferedSqs.receiveMessage(receiveRq);
```

Configurazione di AmazonSQSBufferedAsyncClient

`AmazonSQSBufferedAsyncClient` è preconfigurato con impostazioni che funzionano per la maggior parte dei casi d'uso. È possibile configurare ulteriormente `AmazonSQSBufferedAsyncClient`, ad esempio:

1. Crea un'istanza della classe `QueueBufferConfig` con i parametri di configurazione obbligatori.
2. Fornisci l'istanza al costruttore `AmazonSQSBufferedAsyncClient`.

```
// Create the basic Amazon SQS async client
final AmazonSQSAsync sqsAsync = new AmazonSQSAsyncClient();

final QueueBufferConfig config = new QueueBufferConfig()
    .withMaxInflightReceiveBatches(5)
    .withMaxDoneReceiveBatches(15);

// Create the buffered client
final AmazonSQSAsync bufferedSqs = new AmazonSQSBufferedAsyncClient(sqsAsync, config);
```

Parametri di configurazione QueueBufferConfig

Parametro	Valore predefinito	Descrizione
`longPoll`	`true`	Quando `longPoll` è impostato su `true`, `AmazonSQSBufferedA`

Parametro	Valore predefinito	Descrizione
		`syncClient` tenta di utilizzare il polling lungo durante l'utilizzo dei messaggi.
`longPollWaitTimeoutSeconds`	20 s	Il tempo massimo, in secondi, che una chiamata di `ReceiveMessage` si blocca sul server in attesa che i messaggi compaiano nella coda prima di restituire un risultato di ricezione vuoto. **Note** Questa impostazione non ha alcun effetto se il polling lungo è disattivato.

Parametro	Valore predefinito	Descrizione
`maxBatchOpenMs`	200 ms	Il tempo massimo, in millisecondi, che una chiamata in uscita attende altre chiamate dello stesso tipo con cui raggrupparsi. Maggiore è l'impostazione, minore è il numero di batch necessari per eseguire la stessa quantità di lavoro (tuttavia, la prima chiamata in un batch deve dedicare più tempo all'attesa). Quando questo parametro è impostato su `0`, le richieste inviate non attendono altre richieste, disabilitando effettivamente il batching.

Parametro	Valore predefinito	Descrizione
`maxBatchSize`	10 richieste per batch	Il numero massimo di messaggi raggruppati in una singola richiesta. Maggiore è l'impostazione, minore è il numero di batch che sono necessari per eseguire la stessa quantità di richieste. **Note** 10 richieste per batch è il valore massimo consentito per Amazon SQS.
`maxBatchSizeBytes`	256 KB	Le dimensioni massime del batch di un messaggio, in byte, che il client tenta di inviare ad Amazon SQS . **Note** 256 KB è il valore massimo consentito per Amazon SQS.

Parametro	Valore predefinito	Descrizione
`maxDoneReceiveBatches`	10 batch	Il numero massimo di pre-fetching e archiviazioni di batch `AmazonSQSBufferedAsyncClient` sul lato client. Maggiore è l'impostazione, più richieste di ricezione possono essere soddisfatte senza la necessità di effettuare una chiamata ad Amazon SQS (tuttavia, più messaggi sono pre-recuperati, più tempo rimangono nel buffer, il che significa che il loro timeout visibilità scade). **Note** `0` indica che tutto il pre-recupero dei messaggi è disattivato e i messaggi vengono utilizzati solo su richiesta.

Parametro	Valore predefinito	Descrizione
`maxInflightOutboundBatches`	5 batch	Il numero massimo di batch in uscita attivi che possono essere elaborati contemporaneamente. Maggiore è l'impostazione, più rapidamente possono essere inviati i batch in uscita (in base alle altre quote, ad esempio CPU o larghezza di banda) e più thread possono essere utilizzati da `AmazonSQSBufferedAsyncClient` .

Parametro	Valore predefinito	Descrizione
`maxInflightReceiveBatches`	10 batch	Il numero massimo di batch di ricezione attivi che possono essere elaborati contemporaneamente. Maggiore è l'impostazione e più messaggi possono essere ricevuti (in base alle quote, ad esempio CPU o larghezza di banda) e più thread possono essere utilizzati da `AmazonSQSBufferedAsyncClient` . **Note** `0` indica che tutto il pre-recupero dei messaggi è disattivato e i messaggi vengono utilizzati solo su richiesta.

Parametro	Valore predefinito	Descrizione
`visibilityTimeoutSeconds`	-1	Quando questo parametro è impostato su un valore diverso da zero, questo timeout visibilità sostituisce il timeout visibilità impostato sulla coda dalla quale vengono utilizzati i messaggi. **Note** `-1` indica che per la coda è selezionata l'impostazione predefinita. Non è possibile impostare il timeout visibilità su `0`.

Aumentare il throughput con il dimensionamento orizzontale e il raggruppamento delle operazioni

Le code Amazon SQS possono fornire un throughput molto elevato. Per ulteriori informazioni sulle quote di velocità di trasmissione effettiva, vedere Quote correlate ai messaggi.

Per ottenere un throughput elevato, devi dimensionare orizzontalmente i produttori e i consumatori dei messaggi (ovvero aggiungere ulteriori produttori e consumatori).

Argomenti

- Dimensionamento orizzontale
- Raggruppare le operazioni
- Esempio di utilizzo di Java Working per richieste con operazioni singole e in batch

Dimensionamento orizzontale

Poiché accedi ad Amazon SQS attraverso un protocollo di richieste/risposte HTTP, la latenza della richiesta (l'intervallo di tempo compreso tra l'avvio di una richiesta e la ricezione di una risposta) limita il throughput che puoi ottenere da un singolo thread su un'unica connessione. Ad esempio, se la latenza da un client basato su Amazon EC2 ad Amazon SQS nella stessa regione è in media di 20 ms, la velocità di trasmissione effettiva massima da un singolo thread su un'unica connessione è in media di 50 TPS.

Per dimensionamento orizzontale si intende aumentare il numero di produttori di messaggi (che generano richieste `SendMessage`) e di consumatori di messaggi (che generano richieste `ReceiveMessage` e `DeleteMessage`) per aumentare il throughput complessivo della coda. È possibile dimensionare orizzontalmente in tre modi:

- Aumentare il numero di thread per client
- Aggiungi altri client
- Aumentare il numero di thread per client e aggiungere altri client

Aggiungendo più client dovresti ottenere essenzialmente guadagni lineari nel throughput della coda. Ad esempio, se raddoppi il numero di clienti, puoi ottenere il doppio del throughput.

> **Note**
>
> Quando si ridimensiona in orizzontale, è necessario assicurarsi che il client Amazon SQS disponga di connessioni o thread sufficienti per supportare il numero di produttori e consumatori di messaggi simultanei che inviano richieste e ricevono risposte. Ad esempio, per impostazione predefinita, le istanze della classe AWS SDK for Java `AmazonSQSClient` mantengono al massimo 50 connessioni ad Amazon SQS. Per creare produttori e consumatori simultanei aggiuntivi, devi modificare il numero massimo di thread produttore e consumatore consentiti su un oggetto `AmazonSQSClientBuilder`, ad esempio:
>
> ```
> final AmazonSQS sqsClient = AmazonSQSClientBuilder.standard()
>         .withClientConfiguration(new ClientConfiguration()
>                 .withMaxConnections(producerCount + consumerCount))
>         .build();
> ```
>
> Per `AmazonSQSAsyncClient`, devi inoltre accertarti che sia disponibile un numero sufficiente di thread.

Questo esempio funziona solo per Java v. 1.x.

Raggruppare le operazioni

Il Raggruppamento esegue più lavoro durante ogni round trip al servizio (ad esempio, quando invii più messaggi con una singola richiesta `SendMessageBatch`). Le operazioni Amazon SQS in batch sono `SendMessageBatch`, `DeleteMessageBatch` e `ChangeMessageVisibilityBatch`. Per usufruire del batching senza modificare produttori e consumatori, utilizza Amazon SQS Buffered Asynchronous Client.

> **Note**
>
> Poiché `ReceiveMessage` può elaborare 10 messaggi alla volta, non c'è nessuna azione `ReceiveMessageBatch`.

Il raggruppamento distribuisce la latenza dell'operazione in batch su più messaggi in una richiesta batch, invece di accettare l'intera latenza per un solo messaggio (per esempio, una richiesta `SendMessage`). Poiché ogni round trip trasporta più lavoro, le richieste in batch fanno un utilizzo più efficiente di thread e connessioni , migliorano così il throughput.

Puoi abbinare il batching al dimensionamento orizzontale per offrire un throughput con un numero minore di thread, connessioni e richieste rispetto a quelli necessari per richieste di messaggi individuali. Puoi utilizzare le operazioni in batch di Amazon SQS per inviare, ricevere o eliminare fino a 10 messaggi alla volta. Poiché Amazon SQS addebita un costo per richiesta, il raggruppamento è in grado di ridurre in modo significativo i costi.

Il raggruppamento può introdurre alcune complessità per la tua applicazione (ad esempio, l'applicazione deve accumulare i messaggi prima di inviarli oppure talvolta è necessario attendere di più per una risposta). Tuttavia, il raggruppamento può essere ancora efficace nei seguenti casi:

- La tua applicazione genera una notevole quantità di messaggi in un breve periodo di tempo, pertanto il ritardo non è mai molto lungo.
- Il consumatore di un messaggio recupera i messaggi da una coda a sua discrezione, a differenza dei tipici produttori di messaggi che devono inviare messaggi in risposta a eventi che non controllano.

> **Important**
>
> Una richiesta di batch potrebbe avere esito positivo anche se i singoli messaggi nel batch hanno avuto esito negativo. Dopo una richiesta di batch, verifica sempre la presenza di errori di messaggio individuali e, se necessario, prova nuovamente l'operazione.

Esempio di utilizzo di Java Working per richieste con operazioni singole e in batch

Prerequisiti

Aggiungi i pacchetti `aws-java-sdk-sqs.jar`, `aws-java-sdk-ec2.jare` `commons-logging.jar` al percorso di classe build Java. L'esempio seguente mostra queste dipendenze in un file `pom.xml` di progetto Maven.

```
<dependencies>
    <dependency>
        <groupId>com.amazonaws</groupId>
        <artifactId>aws-java-sdk-sqs</artifactId>
        <version>LATEST</version>
    </dependency>
    <dependency>
        <groupId>com.amazonaws</groupId>
        <artifactId>aws-java-sdk-ec2</artifactId>
        <version>LATEST</version>
    </dependency>
    <dependency>
        <groupId>commons-logging</groupId>
        <artifactId>commons-logging</artifactId>
        <version>LATEST</version>
    </dependency>
</dependencies>
```

SimpleProducerConsumer.java

Il seguenti esempio di codice Java implementa un semplice modello produttore-consumatore. Il thread principale genera dinamicamente diversi thread produttore e consumatore che elaborano messaggi di 1 KB per un periodo di tempo specificato. Questo esempio include produttori e consumatori che effettuano richieste con operazioni singole e altri che effettuano richieste in batch.

```
/*
```

```
 * Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

import com.amazonaws.AmazonClientException;
import com.amazonaws.ClientConfiguration;
import com.amazonaws.services.sqs.AmazonSQS;
import com.amazonaws.services.sqs.AmazonSQSClientBuilder;
import com.amazonaws.services.sqs.model.*;
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;

import java.math.BigInteger;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import java.util.Scanner;
import java.util.concurrent.TimeUnit;
import java.util.concurrent.atomic.AtomicBoolean;
import java.util.concurrent.atomic.AtomicInteger;

/**
 * Start a specified number of producer and consumer threads, and produce-consume
 * for the least of the specified duration and 1 hour. Some messages can be left
 * in the queue because producers and consumers might not be in exact balance.
 */
public class SimpleProducerConsumer {

    // The maximum runtime of the program.
    private final static int MAX_RUNTIME_MINUTES = 60;
    private final static Log log = LogFactory.getLog(SimpleProducerConsumer.class);

    public static void main(String[] args) throws InterruptedException {
```

```
        final Scanner input = new Scanner(System.in);

        System.out.print("Enter the queue name: ");
        final String queueName = input.nextLine();

        System.out.print("Enter the number of producers: ");
        final int producerCount = input.nextInt();

        System.out.print("Enter the number of consumers: ");
        final int consumerCount = input.nextInt();

        System.out.print("Enter the number of messages per batch: ");
        final int batchSize = input.nextInt();

        System.out.print("Enter the message size in bytes: ");
        final int messageSizeByte = input.nextInt();

        System.out.print("Enter the run time in minutes: ");
        final int runTimeMinutes = input.nextInt();

        /*
         * Create a new instance of the builder with all defaults (credentials
         * and region) set automatically. For more information, see Creating
         * Service Clients in the AWS SDK for Java Developer Guide.
         */
        final ClientConfiguration clientConfiguration = new ClientConfiguration()
                .withMaxConnections(producerCount + consumerCount);

        final AmazonSQS sqsClient = AmazonSQSClientBuilder.standard()
                .withClientConfiguration(clientConfiguration)
                .build();

        final String queueUrl = sqsClient
                .getQueueUrl(new GetQueueUrlRequest(queueName)).getQueueUrl();

        // The flag used to stop producer, consumer, and monitor threads.
        final AtomicBoolean stop = new AtomicBoolean(false);

        // Start the producers.
        final AtomicInteger producedCount = new AtomicInteger();
        final Thread[] producers = new Thread[producerCount];
        for (int i = 0; i < producerCount; i++) {
            if (batchSize == 1) {
```

```
                producers[i] = new Producer(sqsClient, queueUrl, messageSizeByte,
                        producedCount, stop);
            } else {
                producers[i] = new BatchProducer(sqsClient, queueUrl, batchSize,
                        messageSizeByte, producedCount,
                        stop);
            }
            producers[i].start();
        }

        // Start the consumers.
        final AtomicInteger consumedCount = new AtomicInteger();
        final Thread[] consumers = new Thread[consumerCount];
        for (int i = 0; i < consumerCount; i++) {
            if (batchSize == 1) {
                consumers[i] = new Consumer(sqsClient, queueUrl, consumedCount,
                        stop);
            } else {
                consumers[i] = new BatchConsumer(sqsClient, queueUrl, batchSize,
                        consumedCount, stop);
            }
            consumers[i].start();
        }

        // Start the monitor thread.
        final Thread monitor = new Monitor(producedCount, consumedCount, stop);
        monitor.start();

        // Wait for the specified amount of time then stop.
        Thread.sleep(TimeUnit.MINUTES.toMillis(Math.min(runTimeMinutes,
                MAX_RUNTIME_MINUTES)));
        stop.set(true);

        // Join all threads.
        for (int i = 0; i < producerCount; i++) {
            producers[i].join();
        }

        for (int i = 0; i < consumerCount; i++) {
            consumers[i].join();
        }

        monitor.interrupt();
        monitor.join();
```

```
    }

    private static String makeRandomString(int sizeByte) {
        final byte[] bs = new byte[(int) Math.ceil(sizeByte * 5 / 8)];
        new Random().nextBytes(bs);
        bs[0] = (byte) ((bs[0] | 64) & 127);
        return new BigInteger(bs).toString(32);
    }

    /**
     * The producer thread uses {@code SendMessage}
     * to send messages until it is stopped.
     */
    private static class Producer extends Thread {
        final AmazonSQS sqsClient;
        final String queueUrl;
        final AtomicInteger producedCount;
        final AtomicBoolean stop;
        final String theMessage;

        Producer(AmazonSQS sqsQueueBuffer, String queueUrl, int messageSizeByte,
                 AtomicInteger producedCount, AtomicBoolean stop) {
            this.sqsClient = sqsQueueBuffer;
            this.queueUrl = queueUrl;
            this.producedCount = producedCount;
            this.stop = stop;
            this.theMessage = makeRandomString(messageSizeByte);
        }

        /*
         * The producedCount object tracks the number of messages produced by
         * all producer threads. If there is an error, the program exits the
         * run() method.
         */
        public void run() {
            try {
                while (!stop.get()) {
                    sqsClient.sendMessage(new SendMessageRequest(queueUrl,
                            theMessage));
                    producedCount.incrementAndGet();
                }
            } catch (AmazonClientException e) {
                /*
                 * By default, AmazonSQSClient retries calls 3 times before
```

```
                 * failing. If this unlikely condition occurs, stop.
                 */
                log.error("Producer: " + e.getMessage());
                System.exit(1);
            }
        }
    }

    /**
     * The producer thread uses {@code SendMessageBatch}
     * to send messages until it is stopped.
     */
    private static class BatchProducer extends Thread {
        final AmazonSQS sqsClient;
        final String queueUrl;
        final int batchSize;
        final AtomicInteger producedCount;
        final AtomicBoolean stop;
        final String theMessage;

        BatchProducer(AmazonSQS sqsQueueBuffer, String queueUrl, int batchSize,
                      int messageSizeByte, AtomicInteger producedCount,
                      AtomicBoolean stop) {
            this.sqsClient = sqsQueueBuffer;
            this.queueUrl = queueUrl;
            this.batchSize = batchSize;
            this.producedCount = producedCount;
            this.stop = stop;
            this.theMessage = makeRandomString(messageSizeByte);
        }

        public void run() {
            try {
                while (!stop.get()) {
                    final SendMessageBatchRequest batchRequest =
                            new SendMessageBatchRequest().withQueueUrl(queueUrl);

                    final List<SendMessageBatchRequestEntry> entries =
                            new ArrayList<SendMessageBatchRequestEntry>();
                    for (int i = 0; i < batchSize; i++)
                        entries.add(new SendMessageBatchRequestEntry()
                                .withId(Integer.toString(i))
                                .withMessageBody(theMessage));
                    batchRequest.setEntries(entries);
```

```
                final SendMessageBatchResult batchResult =
                        sqsClient.sendMessageBatch(batchRequest);
                producedCount.addAndGet(batchResult.getSuccessful().size());

                /*
                 * Because SendMessageBatch can return successfully, but
                 * individual batch items fail, retry the failed batch items.
                 */
                if (!batchResult.getFailed().isEmpty()) {
                    log.warn("Producer: retrying sending "
                            + batchResult.getFailed().size() + " messages");
                    for (int i = 0, n = batchResult.getFailed().size();
                         i < n; i++) {
                        sqsClient.sendMessage(new
                                SendMessageRequest(queueUrl, theMessage));
                        producedCount.incrementAndGet();
                    }
                }
            }
        } catch (AmazonClientException e) {
            /*
             * By default, AmazonSQSClient retries calls 3 times before
             * failing. If this unlikely condition occurs, stop.
             */
            log.error("BatchProducer: " + e.getMessage());
            System.exit(1);
        }
    }
}

/**
 * The consumer thread uses {@code ReceiveMessage} and {@code DeleteMessage}
 * to consume messages until it is stopped.
 */
private static class Consumer extends Thread {
    final AmazonSQS sqsClient;
    final String queueUrl;
    final AtomicInteger consumedCount;
    final AtomicBoolean stop;

    Consumer(AmazonSQS sqsClient, String queueUrl, AtomicInteger consumedCount,
             AtomicBoolean stop) {
        this.sqsClient = sqsClient;
```

```
            this.queueUrl = queueUrl;
            this.consumedCount = consumedCount;
            this.stop = stop;
        }

        /*
         * Each consumer thread receives and deletes messages until the main
         * thread stops the consumer thread. The consumedCount object tracks the
         * number of messages that are consumed by all consumer threads, and the
         * count is logged periodically.
         */
        public void run() {
            try {
                while (!stop.get()) {
                    try {
                        final ReceiveMessageResult result = sqsClient
                                .receiveMessage(new
                                        ReceiveMessageRequest(queueUrl));

                        if (!result.getMessages().isEmpty()) {
                            final Message m = result.getMessages().get(0);
                            sqsClient.deleteMessage(new
                                    DeleteMessageRequest(queueUrl,
                                    m.getReceiptHandle()));
                            consumedCount.incrementAndGet();
                        }
                    } catch (AmazonClientException e) {
                        log.error(e.getMessage());
                    }
                }
            } catch (AmazonClientException e) {
                /*
                 * By default, AmazonSQSClient retries calls 3 times before
                 * failing. If this unlikely condition occurs, stop.
                 */
                log.error("Consumer: " + e.getMessage());
                System.exit(1);
            }
        }
    }

    /**
     * The consumer thread uses {@code ReceiveMessage} and {@code
     * DeleteMessageBatch} to consume messages until it is stopped.
```

```
     */
    private static class BatchConsumer extends Thread {
        final AmazonSQS sqsClient;
        final String queueUrl;
        final int batchSize;
        final AtomicInteger consumedCount;
        final AtomicBoolean stop;

        BatchConsumer(AmazonSQS sqsClient, String queueUrl, int batchSize,
                      AtomicInteger consumedCount, AtomicBoolean stop) {
            this.sqsClient = sqsClient;
            this.queueUrl = queueUrl;
            this.batchSize = batchSize;
            this.consumedCount = consumedCount;
            this.stop = stop;
        }

        public void run() {
            try {
                while (!stop.get()) {
                    final ReceiveMessageResult result = sqsClient
                            .receiveMessage(new ReceiveMessageRequest(queueUrl)
                                    .withMaxNumberOfMessages(batchSize));

                    if (!result.getMessages().isEmpty()) {
                        final List<Message> messages = result.getMessages();
                        final DeleteMessageBatchRequest batchRequest =
                                new DeleteMessageBatchRequest()
                                        .withQueueUrl(queueUrl);

                        final List<DeleteMessageBatchRequestEntry> entries =
                                new ArrayList<DeleteMessageBatchRequestEntry>();
                        for (int i = 0, n = messages.size(); i < n; i++)
                            entries.add(new DeleteMessageBatchRequestEntry()
                                    .withId(Integer.toString(i))
                                    .withReceiptHandle(messages.get(i)
                                            .getReceiptHandle()));
                        batchRequest.setEntries(entries);

                        final DeleteMessageBatchResult batchResult = sqsClient
                                .deleteMessageBatch(batchRequest);
                        consumedCount.addAndGet(batchResult.getSuccessful().size());

                        /*
```

```
                     * Because DeleteMessageBatch can return successfully,
                     * but individual batch items fail, retry the failed
                     * batch items.
                     */
                    if (!batchResult.getFailed().isEmpty()) {
                        final int n = batchResult.getFailed().size();
                        log.warn("Producer: retrying deleting " + n
                                + " messages");
                        for (BatchResultErrorEntry e : batchResult
                                .getFailed()) {

                            sqsClient.deleteMessage(
                                    new DeleteMessageRequest(queueUrl,
                                            messages.get(Integer
                                                    .parseInt(e.getId()))
                                                    .getReceiptHandle()));

                            consumedCount.incrementAndGet();
                        }
                    }
                }
            }
        } catch (AmazonClientException e) {
            /*
             * By default, AmazonSQSClient retries calls 3 times before
             * failing. If this unlikely condition occurs, stop.
             */
            log.error("BatchConsumer: " + e.getMessage());
            System.exit(1);
        }
    }
}

/**
 * This thread prints every second the number of messages produced and
 * consumed so far.
 */
private static class Monitor extends Thread {
    private final AtomicInteger producedCount;
    private final AtomicInteger consumedCount;
    private final AtomicBoolean stop;

    Monitor(AtomicInteger producedCount, AtomicInteger consumedCount,
            AtomicBoolean stop) {
```

```
            this.producedCount = producedCount;
            this.consumedCount = consumedCount;
            this.stop = stop;
        }

        public void run() {
            try {
                while (!stop.get()) {
                    Thread.sleep(1000);
                    log.info("produced messages = " + producedCount.get()
                            + ", consumed messages = " + consumedCount.get());
                }
            } catch (InterruptedException e) {
                // Allow the thread to exit.
            }
        }
    }
}
```

Monitoraggio delle metriche di volume dall'esecuzione di esempio

Amazon SQS genera automaticamente i parametri di volume per i messaggi inviati, ricevuti ed eliminati. Puoi accedere a tali parametri e ad altri tramite la scheda Monitoraggio per la coda o nella console CloudWatch.

> **Note**
>
> Le metriche possono diventare disponibili fino a 15 minuti dopo l'avvio della coda.

Risorse Amazon SQS correlate

La tabella seguente elenca le risorse correlate che possono essere utili durante l'utilizzo di questo servizio.

Risorsa	Descrizione
Documentazione di riferimento delle API di Amazon Simple Queue Service	Descrizione di operazioni, parametri e tipi di dati e un elenco di errori generati dal servizio.
Amazon SQS nel Riferimento ai comandi di AWS CLI	Descrizioni dei comandi AWS CLI che puoi utilizzare per lavorare con le code.
Regioni ed endpoint	Informazioni su regioni ed endpoint di Amazon SQS
Pagina del prodotto	La pagina Web principale che include informazioni su Amazon SQS.
Forum di discussione	Forum basato su community per sviluppatori per la discussione di questioni tecniche correlate ad Amazon SQS.
Informazioni su AWS Premium Support	Pagina Web principale che include le informazioni su AWS Premium Support, un canale di supporto personale a risposta rapida per aiutarti a creare ed eseguire applicazioni in servizi infrastrutturali AWS.

Cronologia della documentazione

Nella tabella seguente sono descritte le modifiche importanti apportate alla Guida per gli sviluppatori di Amazon Simple Queue Service da gennaio 2019. Per ricevere notifiche sugli aggiornamenti della documentazione, è possibile iscriversi al feed RSS.

Le funzionalità del servizio vengono talvolta implementate in modo incrementale nelle regioni AWS in cui tale servizio è disponibile. Aggiorniamo questa documentazione solo per la prima versione. Non forniamo informazioni sulla disponibilità delle regioni e non annunciamo implementazioni successive delle regioni. Per informazioni sulla disponibilità delle funzionalità del servizio nella regione e per sottoscrivere le notifiche sugli aggiornamenti, consulta la pagina Novità di AWS.

Modifica	Descrizione	Data
Protocollo AWS JSON	Effettua richieste API utilizzando il protocollo AWS JSON.	27 luglio 2023
Nuova sezione per descrivere le policy gestite da AWS per Amazon SQS e gli aggiornamenti a tali policy	Amazon SQS ha aggiunto una nuova azione che consente di elencare le attività di spostamento dei messaggi più recenti (fino a 10) in una coda di origine specifica. Questa operazione è associata all'operazione API `ListMessageMoveTasks` .	7 giugno 2023
Reindirizzamento della coda DLQ tramite le API	Configura i redrive delle code DLQ utilizzando le API di Amazon SQS.	7 giugno 2023
ABAC per Amazon SQS	Controllo degli accessi basato su attributi (ABAC) con tag di coda per autorizzazioni di accesso flessibili e scalabili.	10 novembre 2022

Aumenta il limite di velocità di trasmissione effettiva elevata FIFO	Aumento delle quote predefinite per la modalità FIFO ad alta velocità di trasmissione effettiva nelle regioni commerciali, oltre all'ottimizzazione dei documenti FIFO ad alta velocità di trasmissione effettiva.	20 ottobre 2022
La crittografia lato server (SSE) predefinita è disponibile	Crittografia lato server (SSE) con crittografia di proprietà SQS (SSE-SQS) per impostazione predefinita.	26 settembre 2022
È disponibile il supporto per la protezione alternativa confusa di Amazon SQS	La protezione aggiuntiva confusa consente di specificare nuove intestazioni nelle richieste, che vengono verificate in base alle condizioni della policy KMS quando si utilizza l'SSE gestito da Amazon SQS.	29 dicembre 2021
L'SSE gestito è disponibile	SSE gestito da Amazon SQS (SSE-SQS) è una crittografia gestita lato server che utilizza chiavi di crittografia di proprietà di Amazon SQS per proteggere i dati sensibili inviati tramite code di messaggi.	23 novembre 2021
È disponibile il redrive per le code DLQ	Amazon SQS supporta il redrive delle code DLQ per le code standard.	10 novembre 2021

È disponibile una velocità di trasmissione effettiva elevata per i messaggi nelle code FIFO	La velocità di trasmissione effettiva elevata per le code FIFO di Amazon SQS offre un numero maggiore di transazioni al secondo (TPS) per i messaggi nelle code FIFO. Per informazioni sulle quote di velocità di trasmissione effettiva, consulta Quote relative ai messaggi.	27 maggio 2021
È disponibile una velocità di trasmissione effettiva elevata per i messaggi nelle code FIFO nella versione in anteprima	La velocità di trasmissione effettiva elevata per le code FIFO Amazon SQS è in versione di anteprima ed è soggetta a modifiche. Questa funzionalità offre un numero maggiore di transazioni al secondo (TPS) per i messaggi nelle code FIFO. Per informazioni sulle quote di velocità di trasmissione effettiva, consulta Quote relative ai messaggi.	17 dicembre 2020
Nuovo design della console Amazon SQS	Per semplificare i flussi di lavoro di sviluppo e produzione, la console Amazon SQS offre una nuova esperienza utente.	8 luglio 2020
Amazon SQS supporta l'impaginazione per ListQueues e listDeadLetter SourceQueues	È possibile specificare il numero massimo di risultati da restituire da un listDeadLetterSourceQueuesListQueues o da una richiesta.	22 giugno 2020

Amazon SQS supporta i parametri CloudWatch Amazon di 1 minuto in AWS tutte le regioni, ad eccezione AWS GovCloud delle regioni (Stati Uniti)	Il CloudWatch parametro di un minuto per Amazon SQS è disponibile in tutte le regioni, ad eccezione delle regioni. AWS GovCloud (US)	9 gennaio 2020
Amazon SQS supporta parametri da 1 minuto CloudWatch	La CloudWatch metrica di un minuto per Amazon SQS è attualmente disponibile solo nelle seguenti regioni: Stati Uniti orientali (Ohio), Europa (Irlanda), Europa (Stoccolma) e Asia Pacifico (Tokyo).	25 novembre 2019
Sono disponibili trigger AWS Lambda per le code FIFO di Amazon SQS	È possibile configurare i messaggi in ingresso in una coda FIFO per attivare una funzione Lambda.	25 novembre 2019
La crittografia lato server (SSE) per Amazon SQS è disponibile nelle regioni della Cina	SSE per Amazon SQS è disponibile nelle regioni della Cina.	13 novembre 2019
Le code FIFO sono disponibili nella Regione Medio Oriente (Bahrein)	Le code FIFO sono disponibili nella Regione Medio Oriente (Bahrein).	10 ottobre 2019
Gli endpoint Amazon Virtual Private Cloud (Amazon VPC) per Amazon SQS sono disponibili nelle regioni AWS GovCloud (Stati Uniti orientali) e (Stati Uniti occidentali) AWS GovCloud	Puoi inviare messaggi alle code Amazon SQS da Amazon VPC nelle regioni AWS GovCloud (Stati Uniti orientali) e (Stati Uniti occidentali). AWS GovCloud	5 settembre 2019

Amazon SQS consente la risoluzione dei problemi delle code utilizzando AWS X-Ray e gli attributi del sistema di messaggi	È possibile risolvere i problemi dei messaggi che passano attraverso le code Amazon SQS utilizzando X-Ray. Questa versione aggiunge il parametro di richiesta `MessageSystemAttribute` alle operazioni API `SendMessageBatch` e `SendMessage` (consentendo di inviare intestazioni della traccia X-Ray tramite Amazon SQS), l'attributo `AWSTraceHeader` all'operazione API `ReceiveMessage` e il tipo di dati `MessageSystemAttributeValue`.	28 agosto 2019
È possibile contrassegnare con tag le code Amazon SQS al momento della creazione	Puoi utilizzare una singola chiamata API Amazon SQS, una funzione SDK AWS o un comando AWS Command Line Interface (AWS CLI) per creare contemporaneamente una coda e specificarne i suoi tag. Inoltre, Amazon SQS supporta le chiavi `aws:TagKeys` and `aws:RequestTag` AWS Identity and Access Management (IAM).	22 agosto 2019

Temporary Queue Client per Amazon SQS è ora disponibile	Le code temporanee consentono di risparmiare tempo di sviluppo e costi di implementazione quando si utilizzano schemi di messaggi comuni come request-r esponse. È possibile utilizzar e il Temporary Queue Client per creare code temporanee ad alta velocità, convenienti e gestite dalle applicazioni.	25 luglio 2019
SSE per Amazon SQS è disponibile nella regione (Stati Uniti AWS GovCloud orientali)	La crittografia lato server (SSE) per Amazon SQS è disponibile nella regione (Stati Uniti orientali). AWS GovCloud	20 giugno 2019
Le code FIFO sono disponibi li nelle regioni Asia Pacifico (Hong Kong), Cina (Pechino) , (Stati Uniti orientali) e AWS GovCloud (Stati Uniti occidentali) AWS GovCloud	Le code FIFO sono disponibi li nelle regioni di Asia Pacifico (Hong Kong), Cina (Pechino) , (Stati Uniti orientali) e AWS GovCloud (Stati Uniti occidentali). AWS GovCloud	15 maggio 2019
Le policy endpoint Amazon VPC sono disponibili per Amazon SQS	Puoi creare policy di endpoint Amazon VPC per Amazon SQS.	4 aprile 2019
Le code FIFO sono disponibili nelle regioni Europa (Stoccolm a) e Cina (Ningxia)	Le code FIFO sono disponibili nelle regioni Europa (Stoccolm a) e Cina (Ningxia).	14 marzo 2019

Le code FIFO sono disponibili in tutte le regioni in cui è disponibile Amazon SQS	Le code FIFO sono disponibili nelle regioni Stati Uniti orientali (Virginia settentrionale), Stati Uniti orientali (Ohio), Stati Uniti occidentali (California settentrionale), Stati Uniti occidentali (Oregon), Asia Pacifico (Mumbai), Asia Pacifico (Seoul), Asia Pacifico (Singapore), Asia Pacifico (Sydney), Asia Pacifico (Tokyo), Canada (Centrale), Europa (Francoforte), Europa (Irlanda), Europa (Londra), Europa (Parigi) e Sud America (San Paolo).	7 febbraio 2019

Glossario AWS

Per la terminologia AWS più recente, consultare il glossario AWS nella documentazione di riferimento per Glossario AWS.

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.