aws

Panduan Developerr

Amazon Simple Queue Service

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Simple Queue Service: Panduan Developerr

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan antara para pelanggan, atau dengan cara apa pun yang menghina atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan properti dari masing-masing pemilik, yang mungkin berafiliasi, terkait dengan, atau disponsori oleh Amazon, atau tidak.

Table of Contents

Apa itu Layanan Antrian Sederhana Amazon

Amazon Simple Queue Service (Amazon SQS) menawarkan antrian host yang aman, tahan lama, dan tersedia yang memungkinkan Anda mengintegrasikan dan memisahkan sistem dan komponen perangkat lunak terdistribusi. Amazon SQS menawarkan konstruksi umum seperti antrian huruf mati dan tag alokasi biaya. Ini menyediakan API layanan web generik yang dapat Anda akses menggunakan bahasa pemrograman apa pun yang didukung AWS SDK.

Topik

- Manfaat menggunakan Amazon SQS
- Arsitektur Amazon SQS dasar
- Perbedaan antara Amazon SQS, Amazon MQ, dan Amazon SNS

Manfaat menggunakan Amazon SQS

- Keamanan — Anda mengontrol siapa yang dapat mengirim pesan dan menerima pesan dari antrian Amazon SQS. Anda dapat memilih untuk mengirimkan data sensitif dengan melindungi konten pesan dalam antrian menggunakan enkripsi sisi server (SSE) terkelola Amazon SQS default, atau dengan menggunakan kunci SSE kustom yang dikelola di (). AWS Key Management Service AWS KMS
- Daya Tahan — Demi keamanan pesan Anda, Amazon SQS menyimpannya di beberapa server. Antrian standar mendukung pengiriman at-least-once pesan, dan antrian FIFO mendukung pemrosesan pesan yang tepat sekali dan mode throughput tinggi.
- Ketersediaan - Amazon SQS menggunakan infrastruktur redundan untuk menyediakan akses yang sangat bersamaan ke pesan dan ketersediaan tinggi untuk memproduksi dan mengkonsumsi pesan.
- Skalabilitas — Amazon SQS dapat memproses setiap permintaan buffer secara independen, menskalakan secara transparan untuk menangani kenaikan atau lonjakan beban apa pun tanpa instruksi penyediaan apa pun.
- Keandalan - Amazon SQS mengunci pesan Anda selama pemrosesan, sehingga beberapa produsen dapat mengirim dan beberapa konsumen dapat menerima pesan pada saat yang bersamaan.
- Kustomisasi — Antrian Anda tidak harus persis sama — misalnya, Anda dapat mengatur penundaan default pada antrian. Anda dapat menyimpan konten pesan yang lebih besar dari

256 KB menggunakan Amazon Simple Storage Service (Amazon S3) atau Amazon DynamoDB, dengan Amazon SQS memegang pointer ke objek Amazon S3, atau Anda dapat membagi pesan besar menjadi pesan yang lebih kecil.

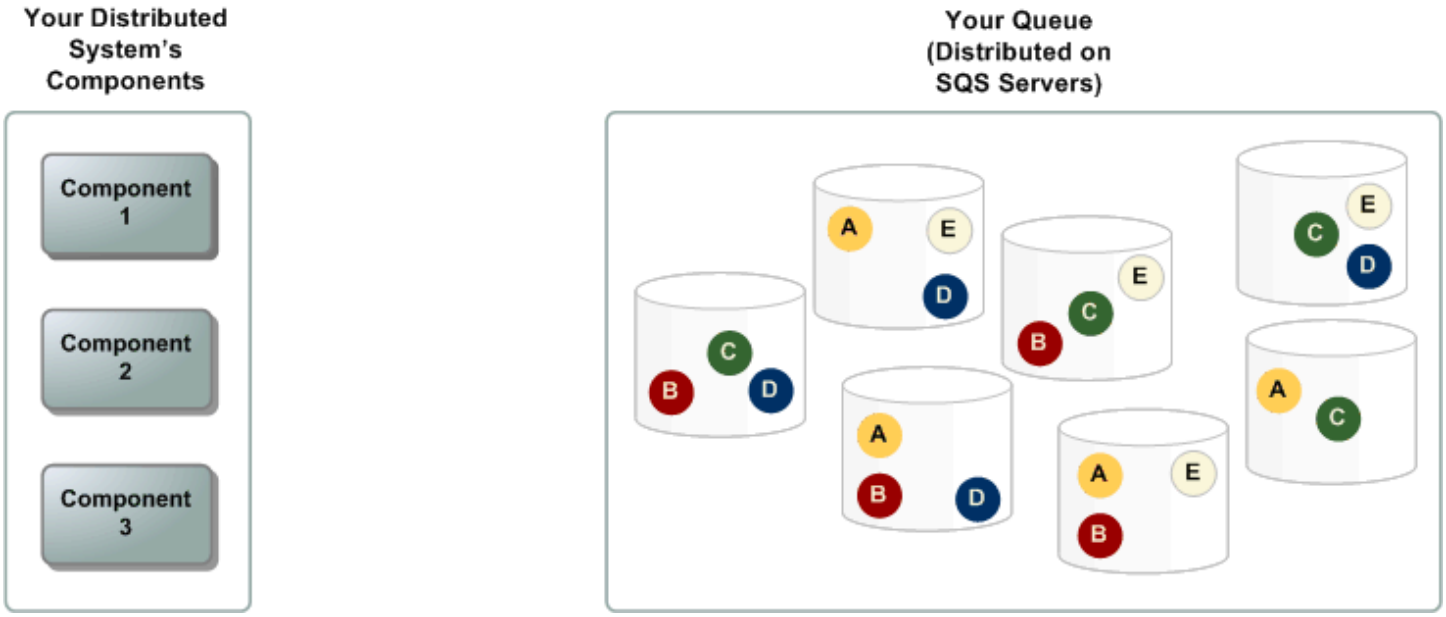
Arsitektur Amazon SQS dasar

Bagian ini menguraikan bagian-bagian dari sistem pesan terdistribusi dan menjelaskan siklus hidup pesan Amazon SQS.

Antrian terdistribusi

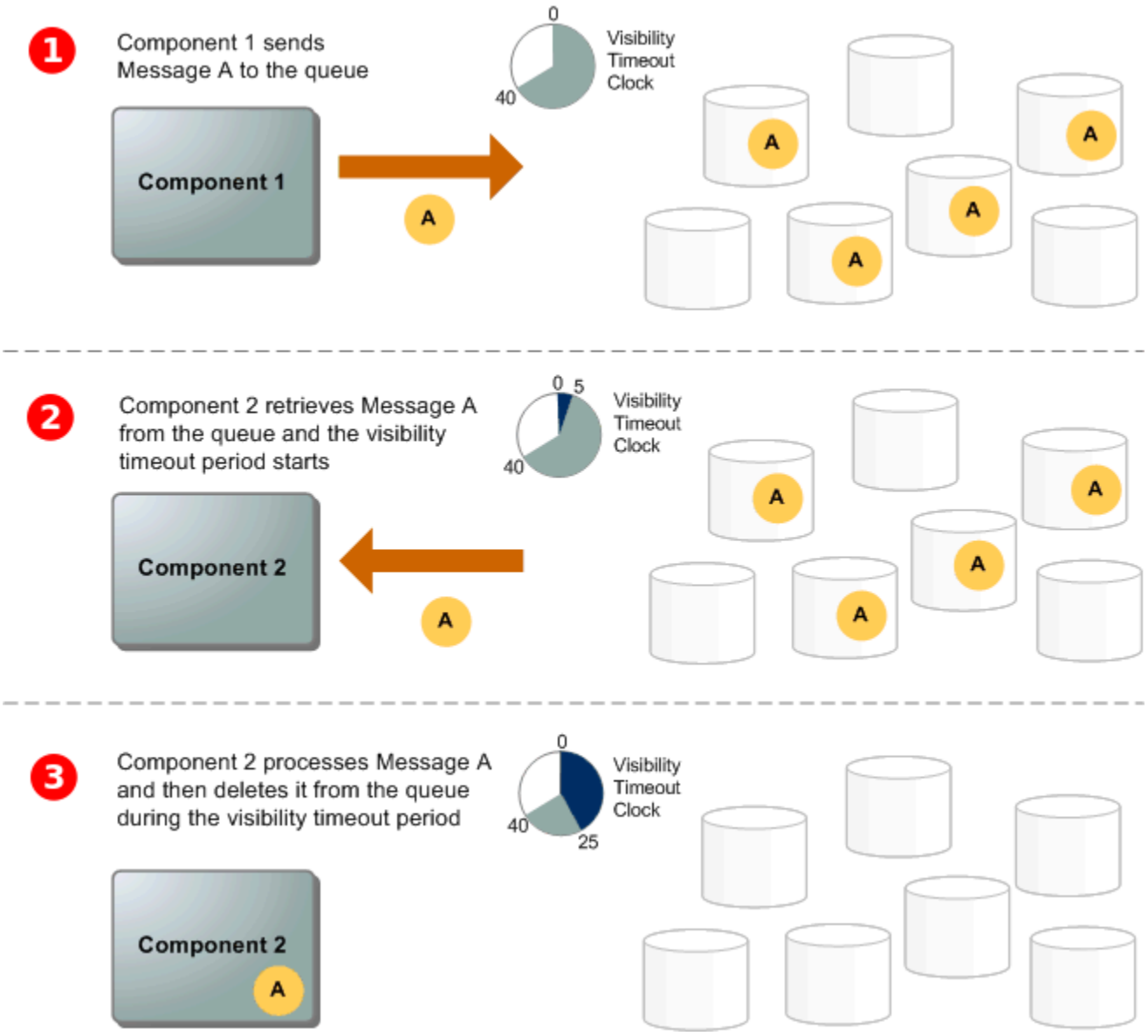
Ada tiga bagian utama dalam sistem pesan terdistribusi: komponen sistem terdistribusi Anda, antrian Anda (didistribusikan di server Amazon SQS), dan pesan dalam antrian.

Dalam skenario berikut, sistem Anda memiliki beberapa produsen (komponen yang mengirim pesan ke antrian) dan konsumen (komponen yang menerima pesan dari antrian). Antrian (yang menyimpan pesan A hingga E) menyimpan pesan secara berlebihan di beberapa server Amazon SQS.



Siklus hidup pesan

Skenario berikut menjelaskan siklus hidup pesan Amazon SQS dalam antrian, dari pembuatan hingga penghapusan.



1 Produs (komponen 1) mengirim pesan A ke antrian, dan pesan didistribusikan ke seluruh server Amazon SQS secara berlebihan.

2 Ketika konsumen (komponen 2) siap untuk memproses pesan, ia mengkonsumsi pesan dari antrian, dan pesan A dikembalikan. Saat pesan A sedang diproses, pesan tetap dalam antrian dan tidak dikembalikan ke permintaan penerimaan berikutnya selama durasi batas waktu visibilitas.

3 Konsu

(komponen 2) menghapus pesan A dari antrian untuk mencegah pesan diterima dan diproses lagi saat batas waktu visibilitas berakhir.

> **Note**
>
> Amazon SQS secara otomatis menghapus pesan yang telah berada dalam antrian selama lebih dari periode penyimpanan pesan maksimum. Periode penyimpanan pesan default adalah 4 hari. Namun, Anda dapat mengatur periode penyimpanan pesan ke nilai dari 60 detik menjadi 1.209.600 detik (14 hari) menggunakan tindakan. `SetQueueAttributes`

Perbedaan antara Amazon SQS, Amazon MQ, dan Amazon SNS

Amazon SQS, Amazon SNS, dan Amazon MQ menawarkan layanan pesan yang sangat terukur easy-to-use dan terkelola, masing-masing dirancang untuk peran tertentu dalam sistem terdistribusi. Berikut adalah ikhtisar yang disempurnakan tentang perbedaan antara layanan ini:

Amazon SQS memisahkan dan menskalakan sistem dan komponen perangkat lunak terdistribusi sebagai layanan antrian. Ini memproses pesan melalui satu pelanggan biasanya, ideal untuk alur kerja di mana pencegahan pesanan dan kerugian sangat penting. Untuk distribusi yang lebih luas, mengintegrasikan Amazon SQS dengan Amazon SNS memungkinkan pola pesan fanout, secara efektif mendorong pesan ke beberapa pelanggan sekaligus.

Amazon SNS memungkinkan penerbit untuk mengirim pesan ke beberapa pelanggan melalui topik, yang berfungsi sebagai saluran komunikasi. Pelanggan menerima pesan yang dipublikasikan menggunakan jenis endpoint yang didukung, seperti Amazon SQS Amazon Data Firehose, Lambda, HTTP, email, notifikasi push seluler, dan pesan teks seluler (SMS). Layanan ini sangat ideal untuk skenario yang membutuhkan pemberitahuan langsung, seperti keterlibatan pengguna waktu nyata atau sistem alarm. Untuk mencegah kehilangan pesan saat pelanggan offline, mengintegrasikan Amazon SNS dengan pesan antrian Amazon SQS memastikan pengiriman yang konsisten.

Amazon MQ paling cocok dengan perusahaan yang ingin bermigrasi dari broker pesan tradisional, mendukung protokol pesan standar seperti AMQP dan MQTT, bersama dengan Apache ActiveMQ dan RabbitMQ. Ini menawarkan kompatibilitas dengan sistem lama yang membutuhkan pesan yang stabil dan andal tanpa konfigurasi ulang yang signifikan.

Bagan berikut memberikan ikhtisar jenis sumber daya masing-masing layanan:

Jenis sumber daya	Amazon SNS	Amazon SQS	Amazon MQ
Sinkron	Tidak	Tidak	Ya
Asinkron	Ya	Ya	Ya
Antrean	Tidak	Ya	Ya
Pesan penerbit-pelanggan	Ya	Tidak	Ya
Broker pesan	Tidak	Tidak	Ya

Baik Amazon SQS dan Amazon SNS direkomendasikan untuk aplikasi baru yang dapat memanfaatkan skalabilitas yang hampir tidak terbatas dan API sederhana. Mereka umumnya menawarkan solusi yang lebih hemat biaya untuk aplikasi volume tinggi dengan harga mereka. pay-as-you-go Kami merekomendasikan Amazon MQ untuk memigrasikan aplikasi dari broker pesan yang ada yang mengandalkan kompatibilitas dengan API seperti JMS atau protokol seperti Advanced Message Queuing Protocol (AMQP), MQTT, dan Simple Text Oriented Message Protocol (STOMP). OpenWire

Menyiapkan Amazon SQS

Sebelum Anda dapat menggunakan Amazon SQS untuk pertama kalinya, Anda harus menyelesaikan langkah-langkah berikut.

Topik

Langkah 1: Buat pengguna Akun AWS dan IAM

Untuk mengakses AWS layanan apa pun, Anda harus terlebih dahulu membuat Akun AWS, akun Amazon.com yang dapat menggunakan AWS produk. Anda dapat menggunakan laporan aktivitas dan penggunaan Anda untuk mengelola autentikasi dan akses. Akun AWS

Untuk menghindari penggunaan pengguna Akun AWS root Anda untuk tindakan Amazon SQS, ini adalah praktik terbaik untuk membuat pengguna IAM untuk setiap orang yang membutuhkan akses administratif ke Amazon SQS.

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/signup.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan tindakan menerima panggilan telepon dan memasukkan kode verifikasi di keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan mengunjungi https://aws.amazon.com/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Langkah 2: Berikan akses terprogram

Untuk menggunakan tindakan Amazon SQS (misalnya, menggunakan Java atau melalui AWS Command Line Interface), Anda memerlukan ID kunci akses dan kunci akses rahasia.

> **Note**
>
> ID kunci akses dan kunci akses rahasia khusus untuk AWS Identity and Access Management. Jangan bingung dengan kredensi untuk AWS layanan lain, seperti pasangan kunci Amazon EC2.

Pengguna membutuhkan akses terprogram jika mereka ingin berinteraksi dengan AWS luar. AWS Management Console Cara untuk memberikan akses terprogram tergantung pada jenis pengguna yang mengakses AWS.

Untuk memberi pengguna akses programatis, pilih salah satu opsi berikut.

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
Identitas tenaga kerja (Pengguna yang dikelola di Pusat Identitas IAM)	Gunakan kredenal sementara untuk menandatangani permintaan terprogram ke AWS CLI, AWS SDK, atau API. AWS	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. • Untuk AWS CLI, lihat Mengkonfigurasi yang akan AWS CLI digunakan AWS IAM Identity Center dalam Panduan AWS Command Line Interface Pengguna. • Untuk AWS SDK, alat, dan AWS API, lihat autentika si Pusat Identitas IAM di Panduan Referensi AWS SDK dan Alat.
IAM	Gunakan kredenal sementara untuk menandatangani permintaan terprogram ke AWS CLI, AWS SDK, atau API. AWS	Mengikuti petunjuk dalam Menggunakan kredensil sementara dengan AWS sumber daya di Panduan Pengguna IAM.
IAM	(Tidak direkomendasikan) Gunakan kredensil jangka panjang untuk menandata ngani permintaan terprogram ke AWS CLI, AWS SDK, atau API. AWS	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. • Untuk mengetahui AWS CLI, lihat Mengautentikasi menggunakan kredensil pengguna IAM di Panduan Pengguna.AWS Command Line Interface

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
		• Untuk AWS SDK dan alat bantu, lihat Mengautentikasi menggunakan kredensi jangka panjang di Panduan Referensi AWS SDK dan Alat. • Untuk AWS API, lihat Mengelola kunci akses untuk pengguna IAM di Panduan Pengguna IAM.

Langkah 3: Bersiaplah untuk menggunakan kode contoh

Panduan ini mencakup contoh yang menggunakan AWS SDK for Java. Untuk menjalankan kode contoh, ikuti petunjuk penyiapan di Memulai dengan AWS SDK for Java 2.0.

Anda dapat mengembangkan AWS aplikasi dalam bahasa pemrograman lain, seperti Go, PythonJavaScript, dan Ruby. Untuk informasi selengkapnya, lihat Alat untuk Dibangun AWS.

> **Note**
>
> Anda dapat menjelajahi Amazon SQS tanpa menulis kode dengan alat seperti AWS Command Line Interface (AWS CLI) atau Windows. PowerShell Anda dapat menemukan AWS CLI contoh di bagian Amazon SQS dari AWS CLI Command Reference. Anda dapat menemukan PowerShell contoh Windows di bagian Layanan Antrian Sederhana Amazon pada Referensi AWS Tools for PowerShell Cmdlet.

Langkah selanjutnya

Anda sekarang siap untuk Memulai mengelola antrian dan pesan Amazon SQS menggunakan file. AWS Management Console

Memulai dengan Amazon SQS

Di bagian ini, Anda akan mempelajari cara membuat antrian standar atau FIFO menggunakan konsol Amazon SQS.

Topik

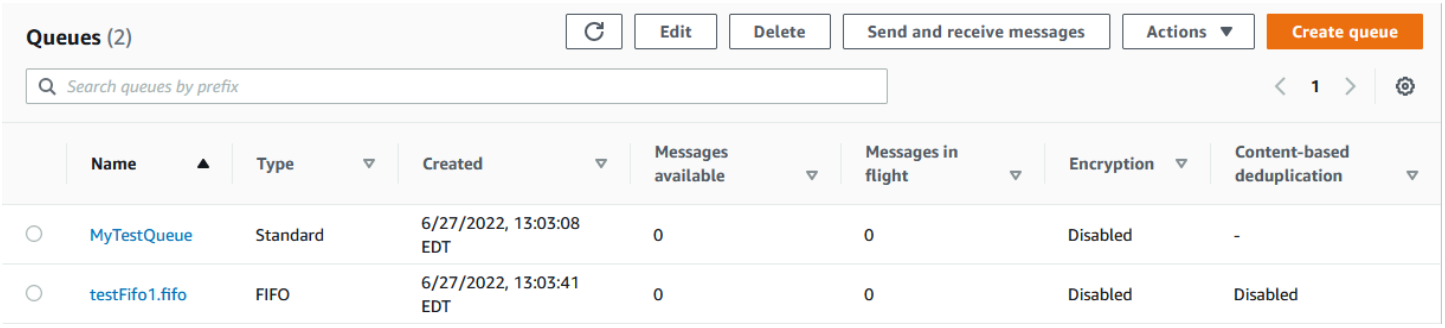
Prasyarat

Sebelum memulai, selesaikan langkah-langkah di Menyiapkan Amazon SQS.

Memahami konsol Amazon SQS

Saat Anda membuka konsol Amazon SQS, pilih Antrian dari panel navigasi. Halaman Antrian memberikan informasi tentang semua antrian Anda di wilayah aktif.

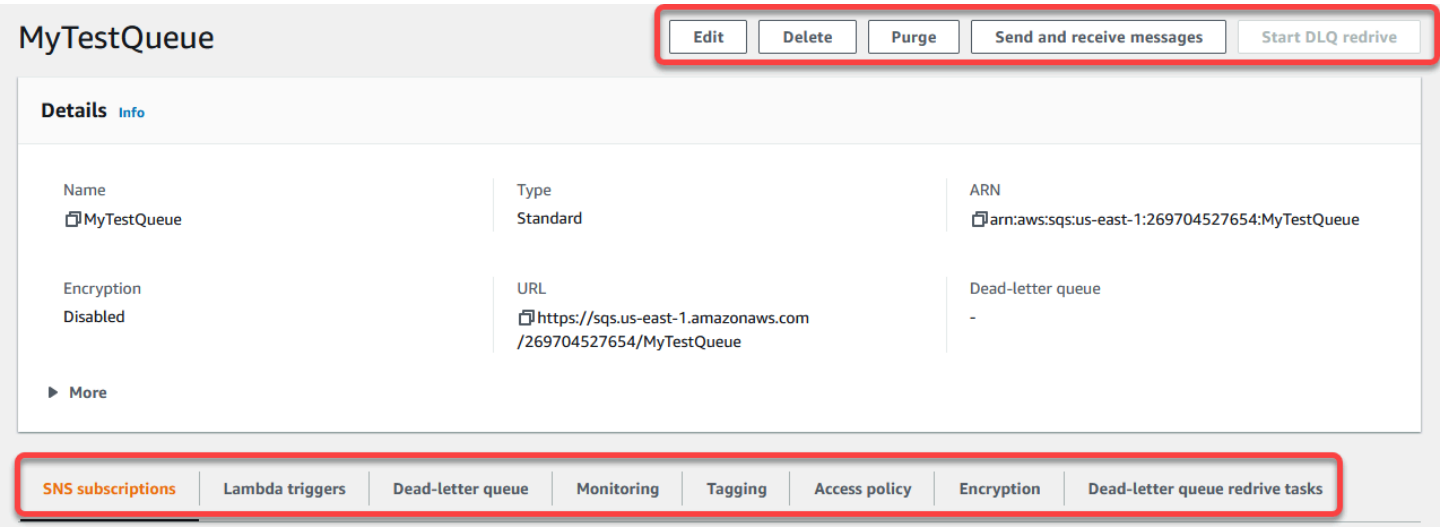
Setiap entri antrian memberikan informasi penting tentang antrian, termasuk tipe dan atribut kuncinya. Antrian standar, dioptimalkan untuk throughput maksimum dan pemesanan pesan upaya terbaik, dibedakan dari antrian First-In-First-Out (FIFO), yang memprioritaskan pemesanan pesan dan keunikan untuk aplikasi yang membutuhkan pengurutan pesan yang ketat.



Elemen dan tindakan interaktif

Dari halaman Antrian, Anda memiliki beberapa opsi untuk mengelola antrian Anda:

1. Tindakan Cepat — Berdekatan dengan setiap nama antrian, menu tarik-turun menawarkan akses cepat ke tindakan umum seperti mengirim pesan, melihat atau menghapus pesan, mengonfigurasi pemicu, dan menghapus antrian itu sendiri.
2. Tampilan dan Konfigurasi Terperinci — Mengklik nama antrian membuka halaman Detailnya, di mana Anda dapat mempelajari lebih dalam pengaturan dan konfigurasi antrian. Di sini, Anda dapat menyesuaikan parameter seperti periode penyimpanan pesan, batas waktu visibilitas, dan ukuran pesan maksimum untuk menyesuaikan antrian dengan persyaratan aplikasi Anda.



Pemilihan wilayah dan tag sumber daya

Pastikan Anda berada di tempat yang benar Wilayah AWS untuk mengakses dan mengelola antrian Anda secara efektif. Selain itu, pertimbangkan untuk menggunakan tag sumber daya untuk mengatur dan mengkategorikan antrian Anda, memungkinkan manajemen sumber daya yang lebih baik, alokasi biaya, dan kontrol akses dalam lingkungan bersama Anda. AWS

Dengan memanfaatkan fitur dan fungsi yang ditawarkan dalam konsol Amazon SQS, Anda dapat mengelola infrastruktur pesan secara efisien, mengoptimalkan kinerja antrian, dan memastikan pengiriman pesan yang andal untuk aplikasi Anda.

Jenis antrian Amazon SQS

Amazon SQS mendukung dua jenis antrian — antrian standar dan antrian FIFO. Gunakan informasi dari tabel berikut untuk memilih antrian yang tepat untuk situasi Anda. Untuk mempelajari selengkapnya tentang antrian Amazon SQS, lihat dan. Memulai dengan antrian standar Amazon SQS Memulai antrian FIFO di Amazon SQS

Antrian standar	Antrian FIFO
Throughput Tanpa Batas — Antrian standar mendukung jumlah panggilan API yang hampir tidak terbatas per detik, per tindakan API (`SendMessage` ,`ReceiveMessage` , atau). `DeleteMessage` Pengiriman At-Least-Once — Pesan dikirim setidaknya sekali, tetapi kadang-kadang lebih dari satu salinan pesan terkirim. Pemesanan Upaya Terbaik - Kadang-kadang, pesan dikirim dalam urutan yang berbeda dari yang dikirim.	Throughput Tinggi — Jika Anda menggunak an batching, antrian FIFO mendukung hingga 3.000 pesan per detik, per metode API (`SendMessageBatch` ,, atau). `ReceiveMe ssage` `DeleteMessageBatch` 3.000 pesan per detik mewakili 300 panggilan API, masing-masing dengan batch 10 pesan. Untuk meminta peningkatan kuota, kirimkan permintaan dukungan. Tanpa batching, antrian FIFO mendukung hingga 300 panggilan API per detik, per metode API (`SendMessage` ,, atau). `ReceiveMessage` `DeleteMessage` Exactly-Once Processing — Pesan dikirim sekali dan tetap tersedia sampai konsumen memproses dan menghapusnya. Duplikat tidak dimasukkan ke dalam antrian. Pengiriman First-In-First-Out - Urutan pengiriman dan penerimaan pesan dipertaha nkan dengan ketat.
	
Kirim data antar aplikasi saat throughput penting, misalnya: • Pisahkan permintaan pengguna langsung dari pekerjaan latar belakang intensif: biarkan pengguna mengunggah media saat mengubah ukuran atau menyandikannya.	Kirim data antar aplikasi saat urutan acara penting, misalnya: • Pastikan perintah yang dimasukkan pengguna dijalankan dalam urutan yang benar. • Tampilkan harga produk yang benar dengan mengirimkan modifikasi harga dalam urutan yang benar.

Antrian standar	Antrian FIFO
• Alokasikan tugas ke beberapa node pekerja: memproses sejumlah besar permintaan validasi kartu kredit. • Pesan batch untuk pemrosesan masa depan: jadwalkan beberapa entri yang akan ditambahkan ke database.	• Cegah siswa mendaftar di kursus sebelum mendaftar untuk akun.

Membuat antrian standar Amazon SQS dan mengirim pesan

Ini adalah cara membuat antrian standar untuk Amazon SQS.

Buat antrian menggunakan konsol Amazon SQS

Anda dapat menggunakan konsol Amazon SQS untuk membuat antrian standar. Konsol menyediakan nilai default untuk semua pengaturan kecuali untuk nama antrian.

> **Important**
>
> Pada 17 Agustus 2022, enkripsi sisi server default (SSE) diterapkan ke semua antrian Amazon SQS.
> Jangan menambahkan informasi identitas pribadi (PII) atau informasi rahasia atau sensitif lainnya dalam nama antrian. Nama antrian dapat diakses oleh banyak Amazon Web Services, termasuk penagihan dan CloudWatch log. Nama antrian tidak dimaksudkan untuk digunakan untuk data pribadi atau sensitif.

Untuk membuat antrean standar Amazon SQS

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Pilih Buat antrean.
3. Untuk Type, tipe antrian Standar diatur secara default.

> **Note**
>
> Anda tidak dapat mengubah jenis antrian setelah Anda membuat antrian.

4. Masukkan Nama untuk antrian Anda.
5. (Opsional) Konsol menetapkan nilai default untuk parameter konfigurasi antrian. Di bawah Konfigurasi, Anda dapat mengatur nilai baru untuk parameter berikut:

   a. Untuk batas waktu Visibilitas, masukkan durasi dan unit. Kisarannya dari 0 detik hingga 12 jam. Nilai defaultnya adalah 30 detik.

   b. Untuk periode penyimpanan Pesan, masukkan durasi dan unit. Kisarannya dari 1 menit hingga 14 hari. Nilai defaultnya adalah 4 hari.

   c. Untuk keterlambatan Pengiriman, masukkan durasi dan unit. Kisarannya dari 0 detik hingga 15 menit. Nilai defaultnya adalah 0 detik.

   d. Untuk Ukuran pesan maksimum, masukkan nilai. Kisarannya dari 1 KB hingga 256 KB. Nilai defaultnya adalah 256 KB.

   e. Untuk Menerima waktu tunggu pesan, masukkan nilai. Kisarannya dari 0 hingga 20 detik. Nilai defaultnya adalah 0 detik, yang menetapkan polling pendek. Setiap nilai bukan nol menetapkan polling panjang.
6. (Opsional) Tentukan kebijakan Akses. Kebijakan akses mendefinisikan akun, pengguna, dan peran yang dapat mengakses antrian. Kebijakan akses juga mendefinisikan tindakan (seperti`SendMessage`,`ReceiveMessage`, atau`DeleteMessage`) yang dapat diakses pengguna. Kebijakan default hanya mengizinkan pemilik antrian untuk mengirim dan menerima pesan.

   Untuk menentukan kebijakan akses, lakukan salah satu hal berikut:

   - Pilih Dasar untuk mengonfigurasi siapa yang dapat mengirim pesan ke antrian dan siapa yang dapat menerima pesan dari antrian. Konsol membuat kebijakan berdasarkan pilihan Anda dan menampilkan kebijakan akses yang dihasilkan di panel JSON hanya-baca.
   - Pilih Advanced untuk mengubah kebijakan akses JSON secara langsung. Ini memungkinkan Anda menentukan serangkaian tindakan khusus yang dapat dilakukan oleh setiap prinsipal (akun, pengguna, atau peran).
7. Untuk kebijakan Izinkan Remrive, pilih Diaktifkan. Pilih salah satu dari berikut ini: Izinkan semua, Dengan antrian, atau Tolak semua. Saat memilih Dengan antrian, tentukan daftar hingga 10 antrian sumber berdasarkan Nama Sumber Daya Amazon (ARN).
8. Amazon SQS menyediakan enkripsi sisi server terkelola secara default. Untuk memilih jenis kunci enkripsi, atau menonaktifkan enkripsi sisi server terkelola Amazon SQS, perluas Enkripsi. Untuk informasi lebih lanjut tentang jenis kunci enkripsi, lihat Mengkonfigurasi enkripsi sisi server

untuk antrian menggunakan kunci enkripsi yang dikelola SQS dan Mengonfigurasi enkripsi sisi server untuk antrian menggunakan konsol Amazon SQS.

> **Note**
>
> Dengan SSE diaktifkan, anonim `SendMessage` dan `ReceiveMessage` permintaan ke antrian terenkripsi akan ditolak. Praktik terbaik keamanan Amazon SQS merekomendasikan agar tidak menggunakan permintaan anonim. Jika Anda ingin mengirim permintaan anonim ke antrian Amazon SQS, pastikan untuk menonaktifkan SSE.

9. (Opsional) Untuk mengonfigurasi antrian surat mati untuk menerima pesan yang tidak terkirim, perluas antrian Dead-letter.
10. (Opsional) Untuk menambahkan tag ke antrian, perluas Tag.
11. Pilih Buat antrean. Amazon SQS membuat antrian dan menampilkan halaman Detail antrian.

Amazon SQS menyebarkan informasi tentang antrian baru di seluruh sistem. Karena Amazon SQS adalah sistem terdistribusi, Anda mungkin mengalami sedikit penundaan sebelum konsol menampilkan antrian di halaman Antrian.

Kirim pesan

Setelah Anda membuat antrian, Anda dapat mengirim pesan ke sana.

1. Dari panel navigasi kiri, pilih Antrian. Dari daftar antrian, pilih antrian yang Anda buat.
2. Dari Tindakan, pilih Kirim dan terima pesan.

   Konsol menampilkan halaman Kirim dan terima pesan.
3. Di badan Pesan, masukkan teks pesan.
4. Untuk antrian standar, Anda dapat memasukkan nilai untuk penundaan Pengiriman dan memilih unit. Misalnya, masukkan `60` dan pilih detik. Untuk informasi selengkapnya, lihat Pengatur waktu pesan Amazon SQS.
5. Pilih Kirim pesan.

   Saat pesan Anda dikirim, konsol menampilkan pesan sukses. Pilih Lihat detail untuk menampilkan informasi tentang pesan terkirim.

Membuat antrian Amazon SQS FIFO dan mengirim pesan

Ini adalah cara membuat antrian FIFO untuk Amazon SQS.

Membuat antrean

Anda dapat menggunakan konsol Amazon SQS untuk membuat antrian FIFO. Konsol menyediakan nilai default untuk semua pengaturan kecuali untuk nama antrian.

> **Important**
>
> Pada 17 Agustus 2022, enkripsi sisi server default (SSE) diterapkan ke semua antrian Amazon SQS.
> Jangan menambahkan informasi identitas pribadi (PII) atau informasi rahasia atau sensitif lainnya dalam nama antrian. Nama antrian dapat diakses oleh banyak Amazon Web Services, termasuk penagihan dan CloudWatch log. Nama antrian tidak dimaksudkan untuk digunakan untuk data pribadi atau sensitif.

Untuk membuat antrean Amazon SQS FIFO

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Pilih Buat antrean.
3. Untuk Type, tipe antrian Standar diatur secara default. Untuk membuat antrian FIFO, pilih FIFO.

   > **Note**
   >
   > Anda tidak dapat mengubah jenis antrian setelah Anda membuat antrian.

4. Masukkan Nama untuk antrian Anda.

   Nama antrian FIFO harus diakhiri dengan akhiran. `.fifo` Akhiran dihitung terhadap kuota nama antrian 80 karakter. Untuk menentukan apakah antrian adalah FIFO, Anda dapat memeriksa apakah nama antrian diakhiri dengan akhiran.

5. (Opsional) Konsol menetapkan nilai default untuk parameter konfigurasi antrian. Di bawah Konfigurasi, Anda dapat mengatur nilai baru untuk parameter berikut:

a. Untuk batas waktu Visibilitas, masukkan durasi dan unit. Kisarannya dari 0 detik hingga 12 jam. Nilai defaultnya adalah 30 detik.

b. Untuk periode penyimpanan Pesan, masukkan durasi dan unit. Kisarannya dari 1 menit hingga 14 hari. Nilai defaultnya adalah 4 hari.

c. Untuk keterlambatan Pengiriman, masukkan durasi dan unit. Kisarannya dari 0 detik hingga 15 menit. Nilai defaultnya adalah 0 detik.

d. Untuk Ukuran pesan maksimum, masukkan nilai. Kisarannya dari 1 KB hingga 256 KB. Nilai defaultnya adalah 256 KB.

e. Untuk Menerima waktu tunggu pesan, masukkan nilai. Kisarannya dari 0 hingga 20 detik. Nilai defaultnya adalah 0 detik, yang menetapkan polling pendek. Setiap nilai bukan nol menetapkan polling panjang.

f. Untuk antrian FIFO, pilih deduplikasi berbasis konten untuk mengaktifkan deduplikasi berbasis konten. Pengaturan default dinonaktifkan.

g. (Opsional) Untuk antrian FIFO untuk mengaktifkan throughput yang lebih tinggi untuk mengirim dan menerima pesan dalam antrian, pilih Aktifkan FIFO throughput tinggi.

   Memilih opsi ini mengubah opsi terkait (cakupan Deduplikasi dan batas throughput FIFO) ke pengaturan yang diperlukan untuk mengaktifkan throughput tinggi untuk antrian FIFO. Jika Anda mengubah salah satu pengaturan yang diperlukan untuk menggunakan FIFO throughput tinggi, throughput normal berlaku untuk antrian, dan deduplikasi terjadi seperti yang ditentukan. Untuk informasi selengkapnya, lihat Throughput tinggi untuk antrian FIFO di Amazon SQS dan Kuota pesan Amazon SQS.

6. (Opsional) Tentukan kebijakan Akses. Kebijakan akses mendefinisikan akun, pengguna, dan peran yang dapat mengakses antrian. Kebijakan akses juga mendefinisikan tindakan (sepertiSendMessage,ReceiveMessage, atauDeleteMessage) yang dapat diakses pengguna. Kebijakan default hanya mengizinkan pemilik antrian untuk mengirim dan menerima pesan.

   Untuk menentukan kebijakan akses, lakukan salah satu hal berikut:

   - Pilih Dasar untuk mengonfigurasi siapa yang dapat mengirim pesan ke antrian dan siapa yang dapat menerima pesan dari antrian. Konsol membuat kebijakan berdasarkan pilihan Anda dan menampilkan kebijakan akses yang dihasilkan di panel JSON hanya-baca.

- Pilih Advanced untuk mengubah kebijakan akses JSON secara langsung. Ini memungkinkan Anda menentukan serangkaian tindakan khusus yang dapat dilakukan oleh setiap prinsipal (akun, pengguna, atau peran).

7. Untuk kebijakan Izinkan Remrive, pilih Diaktifkan. Pilih salah satu dari berikut ini: Izinkan semua, Dengan antrian, atau Tolak semua. Saat memilih Dengan antrian, tentukan daftar hingga 10 antrian sumber berdasarkan Nama Sumber Daya Amazon (ARN).
8. Amazon SQS menyediakan enkripsi sisi server terkelola secara default. Untuk memilih jenis kunci enkripsi, atau menonaktifkan enkripsi sisi server terkelola Amazon SQS, perluas Enkripsi. Untuk informasi lebih lanjut tentang jenis kunci enkripsi, lihat Mengkonfigurasi enkripsi sisi server untuk antrian menggunakan kunci enkripsi yang dikelola SQS danMengonfigurasi enkripsi sisi server untuk antrian menggunakan konsol Amazon SQS.

> **Note**
>
> Dengan SSE diaktifkan, anonim `SendMessage` dan `ReceiveMessage` permintaan ke antrian terenkripsi akan ditolak. Praktik terbaik keamanan Amazon SQS merekomendasikan agar tidak menggunakan permintaan anonim. Jika Anda ingin mengirim permintaan anonim ke antrian Amazon SQS, pastikan untuk menonaktifkan SSE.

9. (Opsional) Untuk mengonfigurasi antrian surat mati untuk menerima pesan yang tidak terkirim, perluas antrian Dead-letter.
10. (Opsional) Untuk menambahkan tag ke antrian, perluas Tag.
11. Pilih Buat antrean. Amazon SQS membuat antrian dan menampilkan halaman Detail antrian.

Amazon SQS menyebarkan informasi tentang antrian baru di seluruh sistem. Karena Amazon SQS adalah sistem terdistribusi, Anda mungkin mengalami sedikit penundaan sebelum konsol menampilkan antrian di halaman Antrian.

Setelah membuat antrian, Anda dapat mengirim pesan ke sana, dan menerima serta menghapus pesan. Anda juga dapat mengedit pengaturan konfigurasi antrian apa pun kecuali jenis antrian.

Kirim pesan

Setelah Anda membuat antrian, Anda dapat mengirim pesan ke sana.

1. Dari panel navigasi kiri, pilih Antrian. Dari daftar antrian, pilih antrian yang Anda buat.

2. Dari Tindakan, pilih Kirim dan terima pesan.

   Konsol menampilkan halaman Kirim dan terima pesan.

3. Di badan Pesan, masukkan teks pesan.
4. Untuk antrean First-In-First-Out (FIFO), masukkan ID grup Pesan. Untuk informasi selengkapnya, lihat Logika pengiriman antrian FIFO di Amazon SQS.
5. (Opsional) Untuk antrian FIFO, Anda dapat memasukkan ID deduplikasi Pesan. Jika Anda mengaktifkan deduplikasi berbasis konten untuk antrian, ID deduplikasi pesan tidak diperlukan. Untuk informasi selengkapnya, lihat Logika pengiriman antrian FIFO di Amazon SQS.
6. Antrian FIFO tidak mendukung pengatur waktu pada pesan individual. Untuk informasi selengkapnya, lihat Pengatur waktu pesan Amazon SQS.
7. Pilih Kirim pesan.

   Saat pesan Anda dikirim, konsol menampilkan pesan sukses. Pilih Lihat detail untuk menampilkan informasi tentang pesan terkirim.

Mengelola antrian Amazon SQS

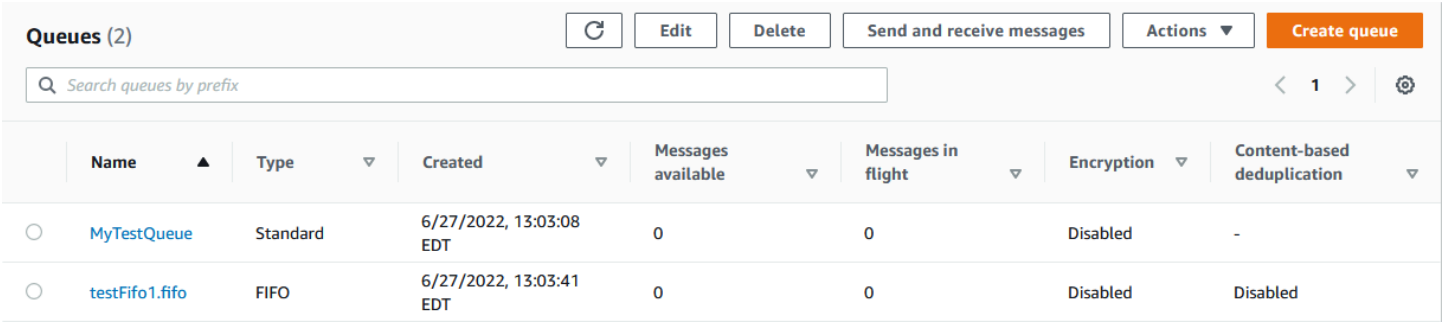
Bagian ini membantu Anda menjadi lebih akrab dengan Amazon SQS dengan menunjukkan cara mengelola antrian dan pesan menggunakan konsol Amazon SQS.

Prasyarat

Sebelum memulai, selesaikan langkah-langkah di Menyiapkan Amazon SQS.

Memahami konsol Amazon SQS

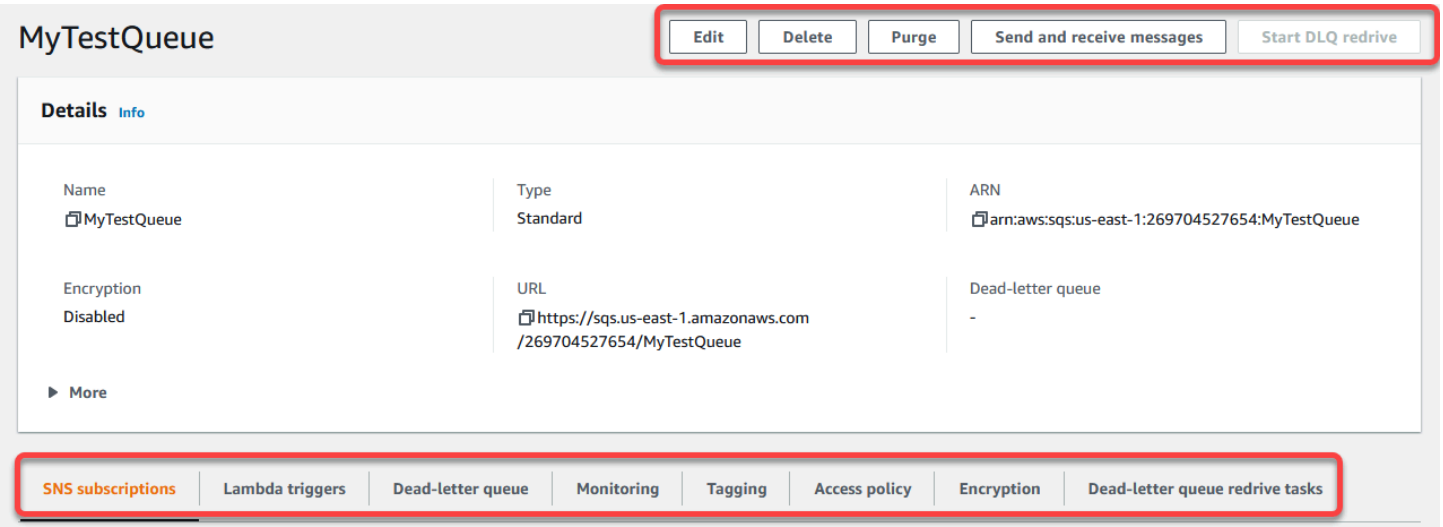
Saat Anda membuka konsol, pilih Antrian dari panel navigasi untuk menampilkan halaman Antrian. Halaman Antrian memberikan informasi tentang semua antrian Anda di wilayah aktif.



Entri untuk setiap antrian menunjukkan jenis antrian dan informasi lain tentang antrian. Kolom Type membantu Anda membedakan antrian standar dari antrian First-In-First Out (FIFO) secara sekilas.

Dari halaman Antrian, ada dua cara untuk melakukan tindakan pada antrian. Anda dapat memilih opsi di sebelah nama antrian dan kemudian memilih tindakan yang ingin Anda lakukan pada antrian.

Anda juga dapat memilih nama antrian, yang membuka halaman Detail untuk antrian. Halaman Detail menyertakan tindakan yang sama dengan halaman Antrian. Selain itu, Anda dapat memilih salah satu tab di bawah bagian Detail untuk melihat detail dan tindakan konfigurasi tambahan.



Mengedit antrian Amazon SQS menggunakan konsol

Anda dapat menggunakan konsol Amazon SQS untuk mengedit parameter konfigurasi antrian apa pun (kecuali jenis antrian) dan menambah atau menghapus fitur antrian.

Untuk mengedit antrian Amazon SQS (konsol)

1. Buka halaman Antrian konsol Amazon SQS.
2. Pilih antrian, lalu pilih Edit.
3. (Opsional) Di bawah Konfigurasi, perbarui parameter konfigurasi antrian.
4. (Opsional) Untuk memperbarui kebijakan akses, di bawah kebijakan Access, ubah kebijakan JSON.
5. (Opsional) Untuk memperbarui kebijakan izin pengaktifan ulang antrian huruf mati, perluas kebijakan Izinkan Remrive.
6. (Opsional) Untuk memperbarui atau menghapus enkripsi, perluas Enkripsi.
7. (Opsional) Untuk menambah, memperbarui, atau menghapus antrean huruf mati (yang memungkinkan Anda menerima pesan yang tidak terkirim), perluas antrian Dead-letter.
8. (Opsional) Untuk menambah, memperbarui, atau menghapus tag untuk antrian, perluas Tag.
9. Pilih Simpan.

   Konsol menampilkan halaman Detail untuk antrian.

Menerima dan menghapus pesan di Amazon SQS

Setelah Anda mengirim pesan ke antrian Amazon SQS, Anda memiliki opsi untuk menerima dan menghapusnya. Saat meminta pesan dari antrian, Anda tidak dapat menentukan pesan individual. Sebagai gantinya, Anda menentukan jumlah maksimum pesan yang ingin Anda ambil, hingga batas 10.

Amazon SQS beroperasi sebagai sistem terdistribusi, yang kadang-kadang dapat menghasilkan respons kosong saat mengambil pesan dari antrian dengan beberapa pesan. Jika ini terjadi, cukup jalankan kembali permintaan Anda. Untuk mengoptimalkan pengambilan pesan dan meminimalkan respons kosong, pertimbangkan untuk menggunakan polling panjang. Polling panjang menunda respons sampai pesan tersedia atau waktu jajak pendapat habis, mengurangi biaya pemungutan suara yang tidak perlu dan meningkatkan efisiensi.

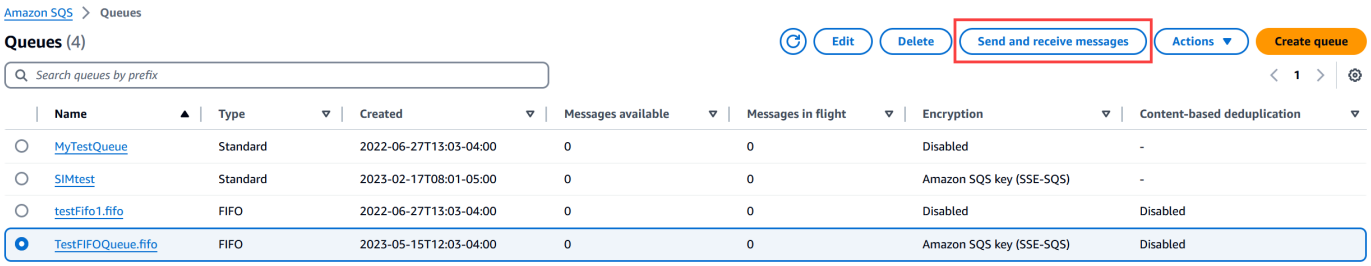
Pesan tidak dihapus secara otomatis setelah pengambilan karena Amazon SQS memastikan bahwa Anda tidak kehilangan akses ke pesan karena kegagalan pemrosesan, seperti masalah dengan aplikasi atau gangguan jaringan. Untuk menghapus pesan secara permanen dari antrian, Anda harus secara eksplisit mengirim permintaan penghapusan setelah memproses pesan untuk mengonfirmasi penerimaan dan penanganan yang berhasil.

Ketika pesan diambil melalui konsol Amazon SQS, mereka segera dibuat terlihat lagi untuk pengambilan ulang. Perilaku default ini memastikan pesan tidak hilang secara tidak sengaja selama operasi manual tetapi dapat menyebabkan pemrosesan berulang. Di lingkungan otomatis, sesuaikan pengaturan batas waktu visibilitas untuk mengontrol berapa lama pesan tetap tidak terlihat oleh konsumen lain setelah diambil. Pengaturan ini sangat penting untuk mengoordinasikan pemrosesan pesan di beberapa konsumen dan memastikan bahwa pesan diproses hanya sekali.

Untuk operasi lebih detail tentang menerima dan menghapus pesan, lihat Panduan Referensi API Amazon SQS. Panduan ini menawarkan informasi komprehensif tentang titik akhir API, termasuk parameter yang mengelola skenario penanganan pesan yang kompleks secara efektif.

Untuk menerima dan menghapus pesan menggunakan konsol

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian.
3. Pada halaman Antrian, pilih antrian, lalu pilih Kirim dan terima pesan.



4. Pada halaman Kirim dan terima pesan, pilih Poll untuk pesan.

   Amazon SQS mulai melakukan polling untuk pesan dalam antrian. Bilah kemajuan di sisi kanan bagian Terima pesan menampilkan durasi polling.

   Bagian Pesan menampilkan daftar pesan yang diterima. Untuk setiap pesan, daftar menampilkan ID pesan, Tanggal terkirim, Ukuran, dan jumlah Terima.
5. Untuk menghapus pesan, pilih pesan yang ingin dihapus, lalu pilih Hapus.
6. Di kotak dialog Hapus Pesan, pilih Hapus.

Mengonfirmasi bahwa antrian Amazon SQS kosong

Dalam kebanyakan kasus, Anda dapat menggunakan polling panjang untuk menentukan apakah antrian kosong. Dalam kasus yang jarang terjadi, Anda mungkin menerima tanggapan kosong bahkan ketika antrean masih berisi pesan, terutama jika Anda menetapkan nilai rendah untuk Menerima pesan waktu tunggu saat Anda membuat antrean. Bagian ini menjelaskan cara mengonfirmasi bahwa antrian kosong.

Untuk mengonfirmasi bahwa antrian kosong (konsol)

1. Hentikan semua produsen mengirim pesan.
2. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
3. Di panel navigasi, pilih Antrian.
4. Pada halaman Antrian, pilih antrian.
5. Pilih tab Pemantauan.
6. Di kanan atas dasbor Pemantauan, pilih panah bawah di sebelah simbol Refresh. Dari menu tarik-turun, pilih Segarkan otomatis. Biarkan interval Refresh pada 1 Menit.
7. Perhatikan dasbor berikut:

   - Perkiraan Jumlah Pesan Tertunda

- Perkiraan Jumlah Pesan Tidak Terlihat
- Perkiraan Jumlah Pesan yang Terlihat

Ketika semuanya menunjukkan `0` nilai selama beberapa menit, antrian kosong.

Untuk mengonfirmasi bahwa antrian kosong (AWS CLI, AWS API)

1. Hentikan semua produsen mengirim pesan.
2. Berulang kali menjalankan salah satu perintah berikut:
   - AWS CLI: `get-queue-attributes`
   - AWS API: `GetQueueAttributes`
3. Perhatikan metrik untuk atribut berikut:
   - `ApproximateNumberOfMessagesDelayed`
   - `ApproximateNumberOfMessagesNotVisible`
   - `ApproximateNumberOfMessagesVisible`

   Ketika semuanya `0` selama beberapa menit, antrian kosong.

Jika Anda mengandalkan CloudWatch metrik Amazon, pastikan Anda melihat beberapa titik data nol berturut-turut sebelum menganggap antrian itu kosong. Untuk informasi selengkapnya tentang CloudWatch metrik, lihat CloudWatch Metrik yang tersedia untuk Amazon SQS.

Menghapus antrian Amazon SQS

Jika Anda tidak lagi menggunakan antrean Amazon SQS dan tidak memperkirakan menggunakannya dalam waktu dekat, kami sarankan untuk menghapusnya.

> **Tip**
>
> Jika Anda ingin memverifikasi bahwa antrian kosong sebelum Anda menghapusnya, lihatMengonfirmasi bahwa antrian Amazon SQS kosong.

Anda dapat menghapus antrian bahkan ketika itu tidak kosong. Untuk menghapus pesan dalam antrian tetapi bukan antrian itu sendiri, bersihkan antrean.

Untuk menghapus antrian (konsol)

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian.
3. Pada halaman Antrian, pilih antrian yang akan dihapus.
4. Pilih Hapus.
5. Di kotak dialog Hapus antrian, konfirmasikan penghapusan dengan memasukkan. **`delete`**
6. Pilih Hapus.

Untuk menghapus antrian (AWS CLI dan API)

Anda dapat menggunakan salah satu perintah berikut untuk menghapus antrian:

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

Membersihkan pesan dari antrian menggunakan konsol Amazon SQS

Jika Anda tidak ingin menghapus antrean Amazon SQS tetapi perlu menghapus semua pesan darinya, bersihkan antrean. Proses penghapusan pesan memakan waktu hingga 60 detik. Kami merekomendasikan menunggu selama 60 detik terlepas dari ukuran antrian Anda.

> **Important**
>
> Saat Anda membersihkan antrian, Anda tidak dapat mengambil pesan yang dihapus.

Untuk membersihkan antrian (konsol)

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian.
3. Pada halaman Antrian, pilih antrian untuk dibersihkan.

4. Dari Actions, pilih Purge.
5. Di kotak dialog Purge queue, konfirmasikan pembersihan dengan memasukkan **`purge`** dan memilih Purge.

   Semua pesan dihapus dari antrian. Konsol menampilkan spanduk konfirmasi.

Tugas umum untuk memulai dengan Amazon SQS

Sekarang setelah Anda membuat antrian dan mempelajari cara mengirim, menerima, dan menghapus pesan dan cara menghapus antrian, Anda mungkin ingin mencoba yang berikut ini:

- Untuk memicu fungsi Lambda, lihat. Mengonfigurasi antrian Amazon SQS untuk memicu fungsi AWS Lambda
- Pelajari cara mengonfigurasi antrian, termasuk SSE dan fitur lainnya.
- Pelajari cara mengirim pesan dengan atribut.
- Pelajari cara mengirim pesan dari VPC.
- Untuk menemukan fungsionalitas dan arsitektur Amazon SQS, lihat Jenis antrian Amazon SQS dan. Arsitektur Amazon SQS dasar
- Untuk mengetahui pedoman dan peringatan yang akan membantu Anda memaksimalkan Amazon SQS, lihat. Praktik terbaik untuk Amazon SQS
- Jelajahi contoh Amazon SQS untuk salah satu AWS SDK, seperti Panduan Pengembang.AWS SDK for Java 2.x
- Untuk mempelajari tentang AWS CLI perintah Amazon SQS, lihat Referensi AWS CLI Perintah.
- Untuk mempelajari tentang tindakan Amazon SQS, lihat Referensi API Layanan Antrian Sederhana Amazon.
- Pelajari cara berinteraksi dengan Amazon SQS secara terprogram: Baca Bekerja dengan API dan jelajahi Pusat Pengembangan:AWS
  - Java
  - JavaScript
  - PHP
  - Python
  - Ruby
  - Windows & .NET

- Pelajari tentang mengawasi biaya dan sumber daya di Memecahkan masalah di Amazon SQS bagian ini.
- Pelajari cara melindungi data Anda dan mengaksesnya di bagian Keamanan.
- Pelajari lebih lanjut tentang alur kerja Amazon SQS di bagian alur kerja proses kontrol akses Amazon SQS.

Memulai dengan antrian standar Amazon SQS

Amazon SQS menawarkan standar sebagai tipe antrian default. Antrian standar mendukung jumlah panggilan API yang hampir tidak terbatas per detik, per tindakan API (`SendMessage`,`ReceiveMessage`, atau`DeleteMessage`). Antrian standar mendukung pengiriman at-least-once pesan. Namun, kadang-kadang (karena arsitektur yang sangat terdistribusi yang memungkinkan throughput hampir tidak terbatas), lebih dari satu salinan pesan mungkin dikirim rusak. Antrian standar menyediakan pemesanan upaya terbaik yang memastikan bahwa pesan umumnya dikirim dalam urutan yang sama seperti yang dikirim.

Amazon SQS secara berlebihan menyimpan pesan di lebih dari satu zona ketersediaan (AZ) sebelum a diakui. `SendMessage` Karena salinan pesan disimpan dalam beberapa AZ, tidak ada satu komputer, jaringan, atau kegagalan AZ yang dapat membuat pesan tidak dapat diakses.

Untuk informasi tentang cara membuat dan mengonfigurasi antrian menggunakan konsol Amazon SQS, lihat. Buat antrian menggunakan konsol Amazon SQS Untuk contoh Java, lihatContoh Amazon SQS Java SDK.

Anda dapat menggunakan antrian pesan standar dalam banyak skenario, selama aplikasi Anda dapat memproses pesan yang tiba lebih dari sekali dan rusak, misalnya:

- Pisahkan permintaan pengguna langsung dari pekerjaan latar belakang intensif - Biarkan pengguna mengunggah media saat mengubah ukuran atau menyandikannya.
- Alokasikan tugas ke beberapa node pekerja — Memproses sejumlah besar permintaan validasi kartu kredit.
- Pesan batch untuk pemrosesan masa depan - Jadwalkan beberapa entri yang akan ditambahkan ke database.

Untuk kuota yang terkait dengan antrian standar, lihat. Kuota

Untuk praktik terbaik bekerja dengan antrian standar, lihat. Rekomendasi untuk standar Amazon SQS dan antrian FIFO

Pemesanan pesan

Antrian standar membuat upaya terbaik untuk menjaga urutan pesan, tetapi lebih dari satu salinan pesan mungkin dikirim rusak. Jika sistem Anda mengharuskan pesanan dipertahankan, sebaiknya

gunakan antrean FIFO (First-In-First-Out) atau menambahkan informasi pengurutan di setiap pesan sehingga Anda dapat menyusun ulang pesan saat diterima.

t-least-once Pengiriman

Amazon SQS menyimpan salinan pesan Anda di beberapa server untuk redundansi dan ketersediaan tinggi. Pada kesempatan yang jarang terjadi, salah satu server yang menyimpan salinan pesan mungkin tidak tersedia saat Anda menerima atau menghapus pesan.

Jika ini terjadi, salinan pesan tidak akan dihapus di server yang tidak tersedia, dan Anda mungkin mendapatkan salinan pesan itu lagi saat menerima pesan. Rancang aplikasi Anda menjadi idempoten (tidak boleh terpengaruh secara negatif saat memproses pesan yang sama lebih dari sekali).

Antrian Amazon SQS dan pengidentifikasi pesan

Bagian ini menjelaskan pengidentifikasi antrian standar dan FIFO. Pengidentifikasi ini dapat membantu Anda menemukan dan memanipulasi antrian dan pesan tertentu.

Pengidentifikasi untuk antrian Standar Amazon SQS

Untuk informasi selengkapnya tentang pengidentifikasi berikut, lihat Referensi API Layanan Antrian Sederhana Amazon.

Nama antrian dan URL

Saat membuat antrian baru, Anda harus menentukan nama antrian yang unik untuk AWS akun dan wilayah Anda. Amazon SQS menetapkan setiap antrian yang Anda buat pengenal yang disebut URL antrian yang menyertakan nama antrian dan komponen Amazon SQS lainnya. Kapan pun Anda ingin melakukan tindakan pada antrian, Anda memberikan URL antreannya.

Berikut ini adalah URL antrian untuk antrian bernama `MyQueue` dimiliki oleh pengguna dengan nomor akun AWS. `123456789012`

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

Anda dapat mengambil URL antrian secara terprogram dengan mencantumkan antrian Anda dan mengurai string yang mengikuti nomor akun. Untuk informasi selengkapnya, lihat `ListQueues`.

ID Pesan

Setiap pesan menerima ID pesan yang ditetapkan sistem yang dikembalikan Amazon SQS kepada Anda dalam respons. `SendMessage` Pengenal ini berguna untuk mengidentifikasi pesan. Panjang maksimum ID pesan adalah 100 karakter.

Pegangan tanda terima

Setiap kali Anda menerima pesan dari antrian, Anda menerima tanda terima untuk pesan tersebut. Pegangan ini dikaitkan dengan tindakan menerima pesan, bukan dengan pesan itu sendiri. Untuk menghapus pesan atau mengubah visibilitas pesan, Anda harus memberikan tanda terima (bukan ID pesan). Dengan demikian, Anda harus selalu menerima pesan sebelum Anda dapat menghapusnya (Anda tidak dapat memasukkan pesan ke dalam antrian dan kemudian mengingatnya). Panjang maksimum pegangan tanda terima adalah 1.024 karakter.

> **Important**
>
> Jika Anda menerima pesan lebih dari sekali, setiap kali Anda menerimanya, Anda mendapatkan pegangan tanda terima yang berbeda. Anda harus memberikan tanda terima yang paling baru diterima saat Anda meminta untuk menghapus pesan (jika tidak, pesan mungkin tidak dihapus).

Berikut ini adalah contoh pegangan tanda terima (rusak di tiga baris).

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

Kuota

Tabel berikut mencantumkan kuota yang terkait dengan antrian standar.

Kuota	Deskripsi
Tunda antrian	Penundaan default (minimum) untuk antrian adalah 0 detik. Maksimal 15 menit.
Antrian terdaftar	1.000 antrian per permintaan. `ListQueues`

Kuota	Deskripsi
Waktu tunggu polling yang panjang	Waktu tunggu polling maksimum yang panjang adalah 20 detik.
Pesan per antrian (backlog)	Jumlah pesan yang dapat disimpan antrian Amazon SQS tidak terbatas.
Pesan per antrian (dalam penerbangan)	Untuk sebagian besar antrian standar (tergantung pada lalu lintas antrian dan backlog pesan), bisa ada maksimum sekitar 120.000 dalam pesan penerbangan (diterima dari antrian oleh konsumen, tetapi belum dihapus dari antrian). Jika Anda mencapai kuota ini saat menggunakan polling singkat, Amazon SQS mengembalikan `OverLimit` pesan kesalahan. Jika Anda menggunakan polling panjang, Amazon SQS tidak mengembalikan pesan kesalahan. Untuk menghindari mencapai kuota, Anda harus menghapus pesan dari antrian setelah diproses. Anda juga dapat meningkatkan jumlah antrian yang Anda gunakan untuk memproses pesan Anda. Untuk meminta peningkatan kuota, kirimkan permintaan dukungan.
Nama antrian	Nama antrian dapat memiliki hingga 80 karakter. Karakter berikut diterima: karakter alfanumerik, tanda hubung (`-`), dan garis bawah (). `_` **Note** Nama antrian peka huruf besar/kecil (misalnya, `Test-queue` dan antrian `test-queue` yang berbeda).
Tag antrian	Kami tidak menyarankan menambahkan lebih dari 50 tag ke antrian. Penandaan mendukung karakter Unicode di UTF-8.

Kuota	Deskripsi
	Tag `Key` diperlukan, tetapi tag `Value` adalah opsional.
	Tag `Key` dan tag `Value` peka huruf besar/kecil.
	Tag `Key` dan tag `Value` dapat menyertakan karakter alfanumerik Unicode di UTF-8 dan spasi putih. Karakter khusus berikut diperbolehkan: `_ . : / = + - @`
	Tag `Key` atau tidak `Value` boleh menyertakan awalan cadangan `aws:` (Anda tidak dapat menghapus kunci tag atau nilai dengan awalan ini).
	`Key`Panjang tag maksimum adalah 128 karakter Unicode di UTF-8. Tag tidak `Key` boleh kosong atau null.
	`Value`Panjang tag maksimum adalah 256 karakter Unicode di UTF-8. Tag `Value` mungkin kosong atau null.
	Tindakan penandaan dibatasi hingga 30 TPS per. Akun AWS Jika aplikasi Anda membutuhkan throughput yang lebih tinggi, kirimkan permintaan.

Memulai antrian FIFO di Amazon SQS

Antrian FIFO (First-In-First-Out) memiliki semua kemampuan antrian standar, tetapi dirancang untuk meningkatkan pesan antar aplikasi ketika urutan operasi dan peristiwa sangat penting, atau di mana duplikat tidak dapat ditoleransi.

Contoh situasi di mana Anda mungkin menggunakan antrian FIFO meliputi:

1. Sistem manajemen pesanan e-Commerce di mana pesanan sangat penting
2. Mengintegrasikan dengan sistem pihak ketiga di mana acara perlu diproses secara berurutan
3. Memproses input yang dimasukkan pengguna dalam urutan yang dimasukkan
4. Komunikasi dan jaringan — Mengirim dan menerima data dan informasi dalam urutan yang sama
5. Sistem komputer — Memastikan bahwa perintah yang dimasukkan pengguna dijalankan dalam urutan yang benar
6. Lembaga pendidikan — Mencegah siswa mendaftar di kursus sebelum mendaftar untuk akun
7. Sistem tiket online — Di mana tiket didistribusikan berdasarkan first come first serve

> **Note**
>
> Antrian FIFO juga menyediakan pemrosesan yang tepat sekali, tetapi memiliki jumlah transaksi per detik (TPS) terbatas. Anda dapat menggunakan mode throughput tinggi Amazon SQS dengan antrian FIFO untuk meningkatkan batas transaksi. Untuk detail tentang penggunaan mode throughput tinggi, lihatThroughput tinggi untuk antrian FIFO di Amazon SQS. Untuk informasi tentang kuota throughput, lihat. the section called “Kuota pesan”

Antrian Amazon SQS FIFO tersedia di semua Wilayah tempat Amazon SQS tersedia.

Untuk informasi lebih lanjut tentang penggunaan antrian FIFO dengan pengurutan kompleks, lihat Memecahkan Tantangan Pemesanan Kompleks dengan Antrian FIFO Amazon SQS.

Untuk informasi tentang cara membuat dan mengonfigurasi antrian menggunakan konsol Amazon SQS, lihat. Buat antrian menggunakan konsol Amazon SQS Untuk contoh Java, lihatContoh Amazon SQS Java SDK.

Untuk praktik terbaik bekerja dengan antrian FIFO, lihat dan. Rekomendasi tambahan untuk antrian Amazon SQS FIFO Rekomendasi untuk standar Amazon SQS dan antrian FIFO

Logika pengiriman antrian FIFO di Amazon SQS

Konsep berikut dapat membantu Anda lebih memahami pengiriman pesan ke dan menerima pesan dari FIFO.

Mengirim pesan

Jika beberapa pesan dikirim berturut-turut ke antrian FIFO, masing-masing dengan ID deduplikasi pesan yang berbeda, Amazon SQS menyimpan pesan dan mengakui transmisi. Kemudian, setiap pesan dapat diterima dan diproses dalam urutan yang tepat di mana pesan dikirim.

Dalam antrian FIFO, pesan diurutkan berdasarkan ID grup pesan. Jika beberapa host (atau thread berbeda pada host yang sama) mengirim pesan dengan ID grup pesan yang sama ke antrian FIFO, Amazon SQS menyimpan pesan dalam urutan kedatangan mereka untuk diproses. Untuk memastikan bahwa Amazon SQS mempertahankan urutan pengiriman dan penerimaan pesan, setiap produsen harus menggunakan ID grup pesan unik untuk mengirim semua pesannya.

Logika antrian FIFO hanya berlaku per ID grup pesan. Setiap ID grup pesan mewakili grup pesan terurut yang berbeda dalam antrean Amazon SQS. Untuk setiap ID grup pesan, semua pesan dikirim dan diterima dalam urutan yang ketat. Namun, pesan dengan nilai ID grup pesan yang berbeda mungkin dikirim dan diterima rusak. Anda harus mengaitkan ID grup pesan dengan pesan. Jika Anda tidak memberikan ID grup pesan, tindakan akan gagal. Jika Anda memerlukan satu grup pesan yang dipesan, berikan ID grup pesan yang sama untuk pesan yang dikirim ke antrean FIFO.

Menerima pesan

Anda tidak dapat meminta untuk menerima pesan dengan ID grup pesan tertentu.

Saat menerima pesan dari antrian FIFO dengan beberapa ID grup pesan, Amazon SQS pertama-tama mencoba mengembalikan sebanyak mungkin pesan dengan ID grup pesan yang sama. Hal ini memungkinkan konsumen lain untuk memproses pesan dengan ID grup pesan yang berbeda. Saat Anda menerima pesan dengan ID grup pesan, tidak ada lagi pesan untuk ID grup pesan yang sama yang dikembalikan kecuali Anda menghapus pesan atau pesan tersebut menjadi terlihat.

> **Note**
>
> Dimungkinkan untuk menerima hingga 10 pesan dalam satu panggilan menggunakan parameter `MaxNumberOfMessages` permintaan `ReceiveMessage` tindakan. Pesan-

pesan ini mempertahankan urutan FIFO mereka dan dapat memiliki ID grup pesan yang sama. Jadi, jika ada kurang dari 10 pesan yang tersedia dengan ID grup pesan yang sama, Anda mungkin menerima pesan dari ID grup pesan lain, dalam kumpulan 10 pesan yang sama, tetapi masih dalam urutan FIFO.

Mencoba lagi beberapa kali

Antrian FIFO memungkinkan produsen atau konsumen untuk mencoba beberapa percobaan ulang:

- Jika produser mendeteksi `SendMessage` tindakan yang gagal, ia dapat mencoba lagi mengirim sebanyak yang diperlukan, menggunakan ID deduplikasi pesan yang sama. Dengan asumsi bahwa produsen menerima setidaknya satu pengakuan sebelum interval deduplikasi berakhir, beberapa percobaan ulang tidak mempengaruhi urutan pesan atau memperkenalkan duplikat.
- Jika konsumen mendeteksi `ReceiveMessage` tindakan yang gagal, ia dapat mencoba lagi sebanyak yang diperlukan, menggunakan ID percobaan permintaan terima yang sama. Dengan asumsi bahwa konsumen menerima setidaknya satu pengakuan sebelum batas waktu visibilitas berakhir, beberapa percobaan ulang tidak memengaruhi urutan pesan.
- Saat Anda menerima pesan dengan ID grup pesan, tidak ada lagi pesan untuk ID grup pesan yang sama yang dikembalikan kecuali Anda menghapus pesan atau pesan tersebut menjadi terlihat.

Pengurutan pesan antrian FIFO di Amazon SQS

Antrian FIFO meningkatkan dan melengkapi antrian standar. Fitur terpenting dari jenis antrian ini adalah pengiriman FIFO (First-In-First-Out) dan pemrosesan tepat sekali:

- Urutan pengiriman dan penerimaan pesan dipertahankan dengan ketat dan pesan dikirimkan sekali dan tetap tidak tersedia sampai konsumen memproses dan menghapusnya.
- Duplikat tidak dimasukkan ke dalam antrian.

Selain itu, antrian FIFO mendukung grup pesan yang memungkinkan beberapa grup pesan yang dipesan dalam satu antrian. Tidak ada kuota untuk jumlah grup pesan dalam antrian FIFO.

Tepat sekali diproses di Amazon SQS

Tidak seperti antrian standar, antrian FIFO tidak memperkenalkan pesan duplikat. Antrian FIFO membantu Anda menghindari pengiriman duplikat ke antrian. Jika Anda mencoba lagi `SendMessage` tindakan dalam interval deduplikasi 5 menit, Amazon SQS tidak memasukkan duplikat apa pun ke dalam antrian.

Untuk mengonfigurasi deduplikasi, Anda harus melakukan salah satu hal berikut:

- Aktifkan deduplikasi berbasis konten. Ini menginstruksikan Amazon SQS untuk menggunakan hash SHA-256 untuk menghasilkan ID deduplikasi pesan menggunakan isi pesan — tetapi bukan atribut pesan. Untuk informasi selengkapnya, lihat dokumentasi tentang`CreateQueue`,`GetQueueAttributes`, dan `SetQueueAttributes` tindakan di Referensi API Layanan Antrian Sederhana Amazon.
- Berikan ID deduplikasi pesan secara eksplisit (atau lihat nomor urut) untuk pesan tersebut. Untuk informasi selengkapnya, lihat dokumentasi tentang`SendMessage`,`SendMessageBatch`, dan `ReceiveMessage` tindakan di Referensi API Layanan Antrian Sederhana Amazon.

Pindah dari antrian standar ke antrian FIFO di Amazon SQS

Jika Anda memiliki aplikasi yang sudah ada yang menggunakan antrian standar dan Anda ingin memanfaatkan fitur pemesanan atau pemrosesan tepat sekali dari antrian FIFO, Anda perlu mengonfigurasi antrian dan aplikasi Anda dengan benar.

> **Note**
>
> Anda tidak dapat mengonversi antrian standar yang ada menjadi antrian FIFO. Untuk melakukan pemindahan, Anda harus membuat antrean FIFO baru untuk aplikasi Anda atau menghapus antrian standar yang ada dan membuatnya kembali sebagai antrian FIFO.

Untuk memastikan bahwa aplikasi Anda berfungsi dengan benar dengan antrian FIFO, gunakan daftar periksa berikut:

- Gunakan mode throughput tinggi yang direkomendasikan untuk FIFO untuk mencapai peningkatan throughput. Untuk mempelajari lebih lanjut tentang kuota perpesanan, lihatKuota pesan Amazon SQS.

- Antrian FIFO tidak mendukung penundaan per pesan, hanya penundaan per antrian. Jika aplikasi Anda menetapkan nilai `DelaySeconds` parameter yang sama pada setiap pesan, Anda harus memodifikasi aplikasi Anda untuk menghapus penundaan per pesan dan mengatur `DelaySeconds` seluruh antrian sebagai gantinya.
- Grup pesan adalah fitur FIFO unik yang memungkinkan pelanggan untuk memproses pesan secara paralel sambil mempertahankan pemesanan masing-masing. Pelanggan mengatur pesan ke dalam grup pesan dengan menentukan ID grup pesan. Grup pesan sering didasarkan pada dimensi bisnis untuk beban kerja tertentu. Untuk meningkatkan skala dengan antrian FIFO, gunakan dimensi bisnis yang lebih terperinci untuk ID pesan. Semakin banyak ID grup pesan yang Anda gunakan untuk mendistribusikan pesan, semakin banyak pesan yang disediakan FIFO untuk dikonsumsi.
- Sebelum mengirim pesan ke antrian FIFO, konfirmasikan hal berikut:
  - Jika aplikasi Anda dapat mengirim pesan dengan badan pesan yang identik, Anda dapat memodifikasi aplikasi Anda untuk memberikan ID deduplikasi pesan unik untuk setiap pesan yang dikirim.
  - Jika aplikasi Anda mengirim pesan dengan badan pesan unik, Anda dapat mengaktifkan deduplikasi berbasis konten.
- Anda tidak perlu membuat perubahan kode apa pun kepada konsumen Anda. Namun, jika membutuhkan waktu lama untuk memproses pesan dan batas waktu visibilitas Anda diatur ke nilai tinggi, pertimbangkan untuk menambahkan ID percobaan permintaan terima ke setiap `ReceiveMessage` tindakan. Ini memungkinkan Anda untuk mencoba lagi menerima upaya jika terjadi kegagalan jaringan dan mencegah antrian berhenti karena gagal menerima upaya.

Untuk informasi selengkapnya, lihat Referensi API Layanan Antrian Sederhana Amazon.

Throughput tinggi untuk antrian FIFO di Amazon SQS

Antrian FIFO throughput tinggi di Amazon SQS secara efisien mengelola throughput pesan tinggi sambil mempertahankan urutan pesan yang ketat, memastikan keandalan dan skalabilitas untuk aplikasi yang memproses banyak pesan. Solusi ini sangat ideal untuk skenario yang menuntut throughput tinggi dan pengiriman pesan yang dipesan.

Antrian FIFO throughput tinggi Amazon SQS tidak diperlukan dalam skenario di mana pemesanan pesan yang ketat tidak penting dan di mana volume pesan yang masuk relatif rendah atau sporadis. Misalnya, jika Anda memiliki aplikasi skala kecil yang memproses pesan yang jarang atau tidak

berurutan, kompleksitas dan biaya tambahan yang terkait dengan antrian FIFO throughput tinggi mungkin tidak dapat dibenarkan. Selain itu, jika aplikasi Anda tidak memerlukan kemampuan throughput yang ditingkatkan yang disediakan oleh antrian FIFO throughput tinggi, memilih antrean Amazon SQS standar mungkin lebih hemat biaya dan lebih mudah dikelola.

Untuk meningkatkan kapasitas permintaan dalam antrian FIFO throughput tinggi, disarankan untuk meningkatkan jumlah grup pesan. Untuk informasi selengkapnya tentang kuota pesan throughput tinggi, lihat kuota layanan Amazon SQS di kuota. Referensi Umum Amazon Web Services

Untuk informasi kuota per antrian dan strategi distribusi data, lihat dan. Kuota pesan Amazon SQS Partisi dan distribusi data untuk throughput tinggi untuk antrian SQS FIFO

Topik

- Kasus penggunaan untuk throughput tinggi untuk antrian Amazon SQS FIFO
- Partisi dan distribusi data untuk throughput tinggi untuk antrian SQS FIFO
- Aktifkan throughput tinggi untuk antrian FIFO di Amazon SQS

Kasus penggunaan untuk throughput tinggi untuk antrian Amazon SQS FIFO

Kasus penggunaan berikut menyoroti beragam aplikasi antrian FIFO throughput tinggi, yang menunjukkan efektivitasnya di seluruh industri dan skenario:

1. Pemrosesan data waktu nyata: Aplikasi yang berurusan dengan aliran data waktu nyata, seperti pemrosesan peristiwa atau konsumsi data telemetri, dapat memperoleh manfaat dari antrian FIFO throughput tinggi untuk menangani masuknya pesan secara terus menerus sambil mempertahankan urutannya untuk analisis yang akurat.
2. Pemrosesan pesanan e-Commerce: Dalam platform e-commerce di mana menjaga urutan transaksi pelanggan sangat penting, antrian FIFO throughput tinggi memastikan bahwa pesanan diproses secara berurutan dan tanpa penundaan, bahkan selama musim belanja puncak.
3. Layanan keuangan: Lembaga keuangan yang menangani perdagangan frekuensi tinggi atau data transaksional mengandalkan Antrian FIFO throughput tinggi untuk memproses data pasar dan transaksi dengan latensi minimal sambil mematuhi persyaratan peraturan yang ketat untuk pemesanan pesan.
4. Streaming media: Platform streaming dan layanan distribusi media memanfaatkan antrian FIFO throughput tinggi untuk mengelola pengiriman file media dan konten streaming, memastikan

pengalaman pemutaran yang lancar bagi pengguna sambil mempertahankan urutan pengiriman konten yang benar.

Partisi dan distribusi data untuk throughput tinggi untuk antrian SQS FIFO

Amazon SQS menyimpan data antrian FIFO di partisi. Partisi adalah alokasi penyimpanan untuk antrian yang secara otomatis direplikasi di beberapa Availability Zone dalam suatu Wilayah. AWS Anda tidak mengelola partisi. Sebagai gantinya, Amazon SQS menangani manajemen partisi.

Untuk antrian FIFO, Amazon SQS memodifikasi jumlah partisi dalam antrian dalam situasi berikut:

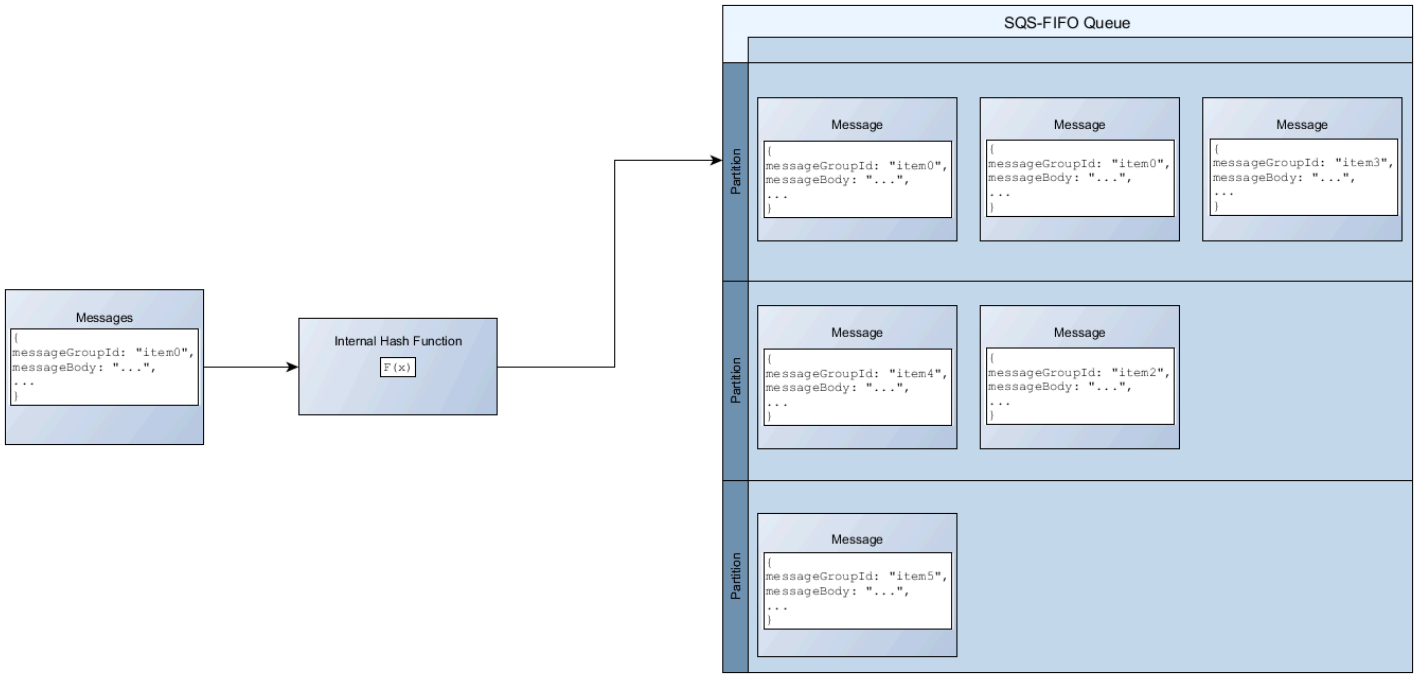
- Jika tingkat permintaan saat ini mendekati atau melebihi apa yang dapat didukung oleh partisi yang ada, partisi tambahan dialokasikan hingga antrian mencapai kuota regional. Untuk informasi tentang kuota, lihatKuota pesan Amazon SQS.
- Jika partisi saat ini memiliki pemanfaatan rendah, jumlah partisi dapat dikurangi.

Manajemen partisi terjadi secara otomatis di latar belakang dan transparan bagi aplikasi Anda. Antrian dan pesan Anda tersedia setiap saat.

Mendistribusikan data dengan ID grup pesan

Untuk menambahkan pesan ke antrian FIFO, Amazon SQS menggunakan nilai ID grup pesan setiap pesan sebagai masukan ke fungsi hash internal. Nilai output dari fungsi hash menentukan partisi mana yang menyimpan pesan.

Diagram berikut menunjukkan antrian yang mencakup beberapa partisi. ID grup pesan antrian didasarkan pada nomor item. Amazon SQS menggunakan fungsi hash untuk menentukan di mana untuk menyimpan item baru; dalam hal ini, itu didasarkan pada nilai hash string. `item0` Perhatikan bahwa item disimpan dalam urutan yang sama di mana mereka ditambahkan ke antrian. Lokasi setiap item ditentukan oleh nilai hash dari ID grup pesannya.



> **Note**
>
> Amazon SQS dioptimalkan untuk distribusi item yang seragam di seluruh partisi antrian FIFO, terlepas dari jumlah partisi. AWS merekomendasikan agar Anda menggunakan ID grup pesan yang dapat memiliki sejumlah besar nilai berbeda.

Mengoptimalkan pemanfaatan partisi

Setiap partisi mendukung hingga 3.000 pesan per detik dengan batching, atau hingga 300 pesan per detik untuk mengirim, menerima, dan menghapus operasi di wilayah yang didukung. Untuk informasi selengkapnya tentang kuota pesan throughput tinggi, lihat kuota layanan Amazon SQS di kuota. Referensi Umum Amazon Web Services

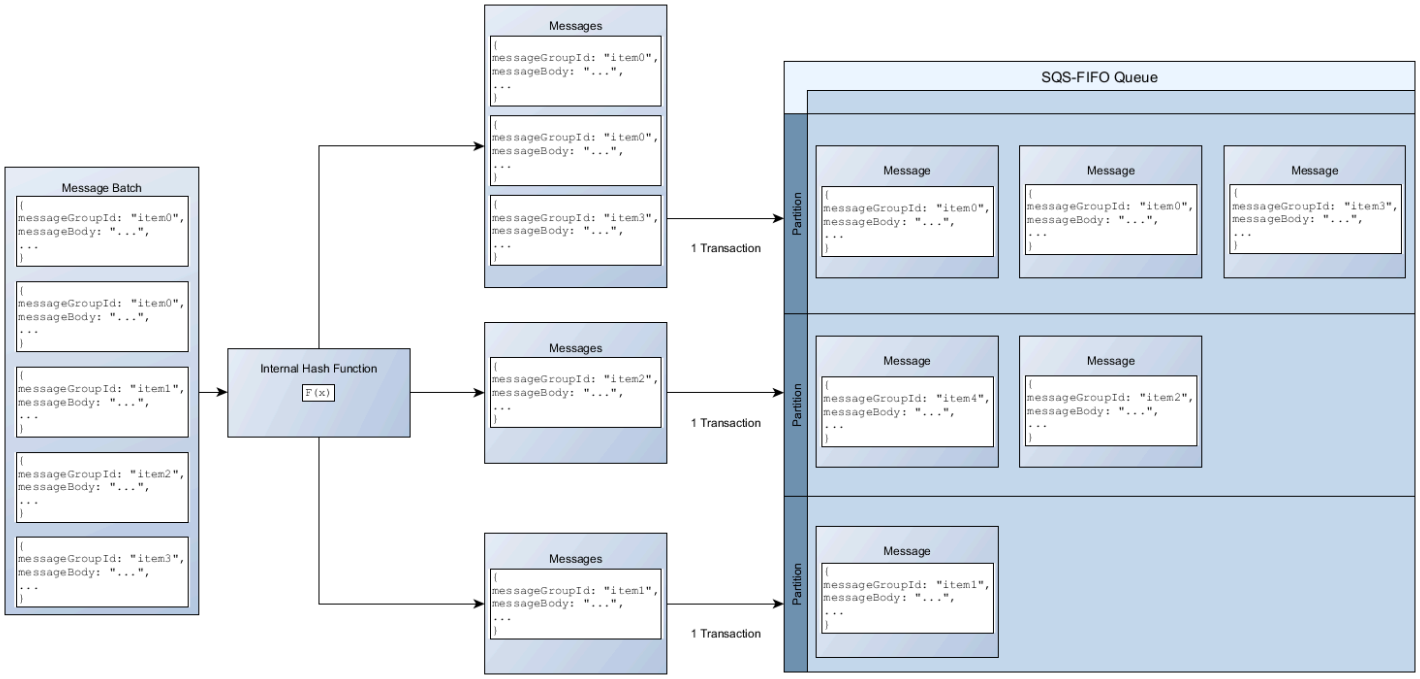
Saat menggunakan API batch, setiap pesan dirutekan berdasarkan proses yang dijelaskan dalamMendistribusikan data dengan ID grup pesan. Pesan yang dialihkan ke partisi yang sama dikelompokkan dan diproses dalam satu transaksi.

Untuk mengoptimalkan pemanfaatan partisi untuk `SendMessageBatch` API, AWS merekomendasikan pengelompokan pesan dengan ID grup pesan yang sama jika memungkinkan.

Untuk mengoptimalkan pemanfaatan partisi untuk `ChangeMessageVisibilityBatch` API `DeleteMessageBatch` dan, AWS rekomendasikan untuk menggunakan `ReceiveMessage`

permintaan dengan `MaxNumberOfMessages` parameter yang disetel ke 10, dan mengelompokkan gagang tanda terima yang dikembalikan oleh satu permintaan. `ReceiveMessage`

Dalam contoh berikut, sekumpulan pesan dengan berbagai ID grup pesan dikirim. Batch dibagi menjadi tiga kelompok, yang masing-masing dihitung terhadap kuota untuk partisi.



> **Note**
>
> Amazon SQS hanya menjamin bahwa pesan dengan fungsi hash internal ID grup pesan yang sama dikelompokkan dalam permintaan batch. Tergantung pada output dari fungsi hash internal dan jumlah partisi, pesan dengan ID grup pesan yang berbeda dapat dikelompokkan. Karena fungsi hash atau jumlah partisi dapat berubah kapan saja, pesan yang dikelompokkan pada satu titik mungkin tidak dikelompokkan nanti.

Aktifkan throughput tinggi untuk antrian FIFO di Amazon SQS

Anda dapat mengaktifkan throughput tinggi untuk antrian FIFO baru atau yang sudah ada. Fitur ini mencakup tiga opsi baru saat Anda membuat dan mengedit antrian FIFO:

- Aktifkan throughput tinggi FIFO — Membuat throughput yang lebih tinggi tersedia untuk pesan dalam antrian FIFO saat ini.
- Lingkup deduplikasi - Menentukan apakah deduplikasi terjadi pada antrian atau tingkat grup pesan.

- Batas throughput FIFO - Menentukan apakah kuota throughput pada pesan dalam antrian FIFO diatur pada tingkat antrian atau grup pesan.

Untuk mengaktifkan throughput tinggi untuk antrian FIFO (konsol)

1. Mulai membuat atau mengedit antrian FIFO.
2. Saat menentukan opsi untuk antrian, pilih Aktifkan FIFO throughput tinggi.

   Mengaktifkan throughput tinggi untuk antrian FIFO menetapkan opsi terkait sebagai berikut:

   - Lingkup deduplikasi diatur ke grup Pesan, pengaturan yang diperlukan untuk menggunakan throughput tinggi untuk antrian FIFO.
   - Batas throughput FIFO diatur ke ID grup Per pesan, pengaturan yang diperlukan untuk menggunakan throughput tinggi untuk antrian FIFO.

   Jika Anda mengubah salah satu pengaturan yang diperlukan untuk menggunakan throughput tinggi untuk antrian FIFO, throughput normal berlaku untuk antrian, dan deduplikasi terjadi seperti yang ditentukan.
3. Lanjutkan menentukan semua opsi untuk antrian. Setelah selesai, pilih Buat antrian atau Simpan.

Setelah membuat atau mengedit antrian FIFO, Anda dapat mengirim pesan ke sana dan menerima dan menghapus pesan, semuanya di TPS yang lebih tinggi. Untuk kuota throughput tinggi, lihat Throughput pesan di. Kuota pesan Amazon SQS

Istilah kunci Amazon SQS

Istilah kunci berikut dapat membantu Anda lebih memahami fungsionalitas antrian FIFO. Untuk informasi selengkapnya, lihat Referensi API Layanan Antrian Sederhana Amazon.

ID deduplikasi pesan

Token yang digunakan untuk deduplikasi pesan terkirim. Jika pesan dengan ID deduplikasi pesan tertentu berhasil dikirim, pesan apa pun yang dikirim dengan ID deduplikasi pesan yang sama berhasil diterima tetapi tidak dikirim selama interval deduplikasi 5 menit.

> **Note**
>
> Amazon SQS terus melacak ID deduplikasi pesan bahkan setelah pesan diterima dan dihapus.

ID grup pesan

Tag yang menentukan bahwa pesan milik grup pesan tertentu. Pesan yang termasuk dalam grup pesan yang sama selalu diproses satu per satu, dalam urutan yang ketat relatif terhadap grup pesan (namun, pesan yang termasuk dalam grup pesan yang berbeda mungkin diproses secara tidak berurutan).

Menerima ID percobaan permintaan

Token yang digunakan untuk deduplikasi panggilan. `ReceiveMessage`

Nomor urut

Jumlah besar dan tidak berurutan yang diberikan Amazon SQS untuk setiap pesan.

Kompatibilitas FIFO di Amazon SQS

Klien

Klien Asinkron Buffered Amazon SQS saat ini tidak mendukung antrian FIFO.

Jasa

Jika aplikasi Anda menggunakan beberapa AWS layanan, atau campuran layanan eksternal, penting untuk memahami fungsionalitas layanan mana yang tidak mendukung antrian FIFO. AWS

Beberapa AWS atau layanan eksternal yang mengirim pemberitahuan ke Amazon SQS mungkin tidak kompatibel dengan antrian FIFO, meskipun memungkinkan Anda untuk menetapkan antrian FIFO sebagai target.

Fitur AWS layanan berikut saat ini tidak kompatibel dengan antrian FIFO:

- Pemberitahuan Acara Amazon S3
- Kait Siklus Hidup Auto Scaling
- AWS IoT Tindakan Aturan

- AWS Lambda Antrian Surat Mati

Untuk informasi tentang kompatibilitas layanan lain dengan antrian FIFO, lihat dokumentasi layanan Anda.

Antrian FIFO dan pengidentifikasi pesan di Amazon SQS

Bagian ini menjelaskan pengidentifikasi antrian FIFO. Pengidentifikasi ini dapat membantu Anda menemukan dan memanipulasi antrian dan pesan tertentu.

Topik

- Pengidentifikasi untuk antrian FIFO di Amazon SQS
- Pengidentifikasi tambahan untuk antrian Amazon SQS FIFO

Pengidentifikasi untuk antrian FIFO di Amazon SQS

Untuk informasi selengkapnya tentang pengidentifikasi berikut, lihat Referensi API Layanan Antrian Sederhana Amazon.

Nama antrian dan URL

Saat membuat antrian baru, Anda harus menentukan nama antrian yang unik untuk AWS akun dan wilayah Anda. Amazon SQS menetapkan setiap antrian yang Anda buat pengenal yang disebut URL antrian yang menyertakan nama antrian dan komponen Amazon SQS lainnya. Kapan pun Anda ingin melakukan tindakan pada antrian, Anda memberikan URL antreannya.

Nama antrian FIFO harus diakhiri dengan akhiran. `.fifo` Akhiran dihitung terhadap kuota nama antrian 80 karakter. Untuk menentukan apakah antrian adalah FIFO, Anda dapat memeriksa apakah nama antrian diakhiri dengan akhiran.

Berikut ini adalah URL antrian untuk antrian FIFO bernama `MyQueue` dimiliki oleh pengguna dengan nomor akun AWS. `123456789012`

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

Anda dapat mengambil URL antrian secara terprogram dengan mencantumkan antrian Anda dan mengurai string yang mengikuti nomor akun. Untuk informasi selengkapnya, lihat `ListQueues`.

ID Pesan

Setiap pesan menerima ID pesan yang ditetapkan sistem yang dikembalikan Amazon SQS kepada Anda dalam respons. `SendMessage` Pengenal ini berguna untuk mengidentifikasi pesan. Panjang maksimum ID pesan adalah 100 karakter.

Pegangan tanda terima

Setiap kali Anda menerima pesan dari antrian, Anda menerima tanda terima untuk pesan tersebut. Pegangan ini dikaitkan dengan tindakan menerima pesan, bukan dengan pesan itu sendiri. Untuk menghapus pesan atau mengubah visibilitas pesan, Anda harus memberikan tanda terima (bukan ID pesan). Dengan demikian, Anda harus selalu menerima pesan sebelum Anda dapat menghapusnya (Anda tidak dapat memasukkan pesan ke dalam antrian dan kemudian mengingatnya). Panjang maksimum pegangan tanda terima adalah 1.024 karakter.

> **Important**
>
> Jika Anda menerima pesan lebih dari sekali, setiap kali Anda menerimanya, Anda mendapatkan pegangan tanda terima yang berbeda. Anda harus memberikan tanda terima yang paling baru diterima saat Anda meminta untuk menghapus pesan (jika tidak, pesan mungkin tidak dihapus).

Berikut ini adalah contoh pegangan tanda terima (rusak di tiga baris).

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

Pengidentifikasi tambahan untuk antrian Amazon SQS FIFO

Untuk informasi selengkapnya tentang pengidentifikasi berikut, lihat Tepat sekali diproses di Amazon SQS Referensi API Layanan Antrian Sederhana Amazon.

ID deduplikasi pesan

Token yang digunakan untuk deduplikasi pesan terkirim. Jika pesan dengan ID deduplikasi pesan tertentu berhasil dikirim, pesan apa pun yang dikirim dengan ID deduplikasi pesan yang sama berhasil diterima tetapi tidak dikirim selama interval deduplikasi 5 menit.

ID grup pesan

Tag yang menentukan bahwa pesan milik grup pesan tertentu. Pesan yang termasuk dalam grup pesan yang sama selalu diproses satu per satu, dalam urutan yang ketat relatif terhadap grup pesan (namun, pesan yang termasuk dalam grup pesan yang berbeda mungkin diproses secara tidak berurutan).

Nomor urut

Jumlah besar dan tidak berurutan yang diberikan Amazon SQS untuk setiap pesan.

Kuota Amazon SQS

Topik ini mencantumkan kuota dalam Amazon Simple Queue Service (Amazon SQS).

Topik

Kuota antrian Amazon SQS FIFO

Kuota Amazon SQS

Tabel berikut mencantumkan kuota yang terkait dengan antrian FIFO.

Kuota	Deskripsi
Tunda antrian	Penundaan default (minimum) untuk antrian adalah 0 detik. Maksimal 15 menit.
Antrian terdaftar	1.000 antrian per permintaan. `ListQueues`
Waktu tunggu polling yang panjang	Waktu tunggu polling maksimum yang panjang adalah 20 detik.
Grup pesan	Tidak ada kuota untuk jumlah grup pesan dalam antrian FIFO.
Pesan per antrian (backlog)	Jumlah pesan yang dapat disimpan antrian Amazon SQS tidak terbatas.
Pesan per antrian (dalam penerbangan)	Untuk antrian FIFO, bisa ada maksimum 20.000 dalam pesan penerbangan (diterima dari antrian oleh konsumen, tetapi belum dihapus dari antrian). Jika Anda mencapai kuota ini, Amazon SQS tidak mengembalikan pesan kesalahan.

Kuota	Deskripsi
Nama antrian	Nama antrian FIFO harus diakhiri dengan akhiran. `.fifo` Akhiran dihitung terhadap kuota nama antrian 80 karakter. Untuk menentukan apakah antrian adalah FIFO, Anda dapat memeriksa apakah nama antrian diakhiri dengan akhiran.
Tag antrian	Kami tidak menyarankan menambahkan lebih dari 50 tag ke antrian. Penandaan mendukung karakter Unicode di UTF-8.
	Tag `Key` diperlukan, tetapi tag `Value` adalah opsional.
	Tag `Key` dan tag `Value` peka huruf besar/kecil.
	Tag `Key` dan tag `Value` dapat menyertakan karakter alfanumerik Unicode di UTF-8 dan spasi putih. Karakter khusus berikut diperbolehkan: `_ . : / = + - @`
	Tag `Key` atau tidak `Value` boleh menyertakan awalan cadangan `aws:` (Anda tidak dapat menghapus kunci tag atau nilai dengan awalan ini).
	`Key`Panjang tag maksimum adalah 128 karakter Unicode di UTF-8. Tag tidak `Key` boleh kosong atau null.
	`Value`Panjang tag maksimum adalah 256 karakter Unicode di UTF-8. Tag `Value` mungkin kosong atau null.
	Tindakan penandaan dibatasi hingga 30 TPS per. Akun AWS Jika aplikasi Anda membutuhkan throughput yang lebih tinggi, kirimkan permintaan.

Kuota pesan Amazon SQS

Tabel berikut mencantumkan kuota yang terkait dengan pesan.

Kuota	Deskripsi
ID pesan batch	ID pesan batch dapat memiliki hingga 80 karakter. Karakter berikut diterima: karakter alfanumerik, tanda hubung (-), dan garis bawah (). _
Atribut pesan	Sebuah pesan dapat berisi hingga 10 atribut metadata.
Batch pesan	Permintaan batch pesan tunggal dapat mencakup maksimal 10 pesan. Untuk informasi selengkapnya, lihat Mengkonfigurasi Klien BufferedAsync AmazonSQS di bagian Tindakan batch Amazon SQS.
Konten pesan	Pesan hanya dapat menyertakan XHTML, JSON, dan teks yang tidak diformat. Karakter Unicode berikut diperbolehkan: #x9 #xA | #xD | | #x20 ke #xD7FF #xFFFD | #xE000 #x10000 ke #x10FFFF Karakter apa pun yang tidak termasuk dalam daftar ini ditolak. Untuk informasi lebih lanjut, lihat spesifikasi W3C untuk karakter.
ID grup pesan	Gunakan pesan dari backlog untuk menghindari membangun backlog besar pesan dengan ID grup pesan yang sama. MessageGroupId diperlukan untuk antrian FIFO. Anda tidak dapat menggunakannya untuk antrian Standar. Anda harus mengaitkan yang tidak kosong MessageGroupId dengan pesan. Jika Anda tidak memberika nMessageGroupId , tindakan gagal. MessageGroupId Panjangnya 128 karakter. Nilai yang valid: karakter alfanumerik dan tanda baca. (!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~)

Kuota	Deskripsi
Retensi pesan	Secara default, pesan disimpan selama 4 hari. Minimal adalah 60 detik (1 menit). Maksimum adalah 1.209.600 detik (14 hari).
Throughput pesan	Antrian standar mendukung jumlah panggilan API yang hampir tidak terbatas per detik, per tindakan API (`SendMessage` ,`ReceiveMessage` , atau`DeleteMessage`).
	Antrian FIFO • Antrian FIFO mendukung kuota 300 transaksi per detik, per tindakan API (`SendMessage` ,, dan). `ReceiveMessage` `DeleteMessage` • Jika Anda menggunakan batching, antrian FIFO mendukung hingga 3.000 pesan per detik, per tindakan API (`SendMessage` ,, dan). `ReceiveMessage` `DeleteMessage` 3.000 pesan per detik mewakili 300 panggilan API, masing-masing dengan batch 10 pesan.

Kuota	Deskripsi
	Throughput tinggi untuk antrian FIFO • Tanpa batching (`SendMessage` ,`ReceiveMessage` , dan`DeleteMessage`), throughput tinggi untuk antrian FIFO memproses hingga 70.000 transaksi per detik, per aksi API di Wilayah AS Timur (Virginia N.), AS Barat (Oregon), dan Eropa (Irlandia). • Untuk Wilayah AS Timur (Ohio) dan Eropa (Frankfur t), throughput default adalah 18.000 transaksi per detik per aksi API. • Untuk wilayah Asia Pasifik (Mumbai), Asia Pasifik (Singapura), Asia Pasifik (Sydney) dan Asia Pasifik (Tokyo), throughput default adalah 9.000 transaksi per detik per aksi API. • Untuk Eropa (London) dan Amerika Selatan (São Paulo), throughput default adalah 4.500 transaksi per detik per aksi API. • Untuk throughput maksimum, tingkatkan jumlah ID grup pesan yang Anda gunakan untuk pesan yang dikirim tanpa pengelompokan. • Anda dapat meningkatkan throughput hingga 700.000 pesan per detik dengan menggunakan API batching (`SendMessageBatch` dan`DeleteMessageBatch`) di Wilayah AS Timur (Virginia N.), AS Barat (Oregon), dan Eropa (Irlandia). 700.000 pesan per detik mewakili 70.000 transaksi per detik, masing-masing dengan batch 10 pesan. Untuk Wilayah Eropa (Frankfurt) dan Timur AS (Ohio), Anda dapat mencapai hingga 180.000 pesan per detik dengan menggunakan API batching. 180.000 pesan per detik mewakili 18.000 transaksi per detik, masing-masing dengan batch 10 pesan.

Kuota	Deskripsi
	Untuk Wilayah Asia Pasifik (Mumbai), Asia Pasifik (Singapura), Asia Pasifik (Sydney), dan Asia Pasifik (Tokyo), Anda dapat mencapai hingga 90.000 pesan per detik dengan batching. Untuk mencapai throughput maksimum saat menggunakan `SendMessageBatch` dan`DeleteMessageBatch` , semua pesan dalam permintaan batch harus menggunakan ID grup pesan yang sama. • Untuk Eropa (London) dan Amerika Selatan (São Paulo), wilayah, Anda dapat mencapai hingga 45.000 pesan per detik dengan batching. Untuk mencapai throughput maksimum saat menggunakan `SendMessageBatch` dan`DeleteMessageBatch` , semua pesan dalam permintaan batch harus menggunakan ID grup pesan yang sama. • Di semua AWS Wilayah lainnya, throughput maksimum adalah 2.400 (tanpa batching) atau 24.000 (menggunakan batching) pesan per detik, per tindakan API. • Untuk meminta kenaikan kuota di atas batas wilayah, kirimkan permintaan dukungan. • Untuk informasi selengkapnya, lihat Partisi dan distribusi data untuk throughput tinggi untuk antrian SQS FIFO.
Pengatur waktu pesan	Penundaan default (minimum) untuk pesan adalah 0 detik. Maksimal 15 menit.

Kuota	Deskripsi
Ukuran pesan	Ukuran pesan minimum adalah 1 byte (1 karakter). Maksimum adalah 262.144 byte (256 KiB). Untuk mengirim pesan yang lebih besar dari 256 KiB, Anda dapat menggunakan Amazon SQS Extended Client Library for Java dan Amazon SQS Extended Client Library untuk Python. Pustaka ini memungkinkan Anda mengirim pesan Amazon SQS yang berisi referensi ke muatan pesan di Amazon S3. Ukuran muatan maksimum adalah 2 GB. **Note** Pustaka yang diperluas ini hanya berfungsi untuk klien sinkron.
Batas waktu visibilitas pesan	Batas waktu visibilitas default untuk pesan adalah 30 detik. Minimal adalah 0 detik. Maksimal 12 jam.
Informasi kebijakan	Kuota maksimum adalah 8.192 byte, 20 pernyataan, 50 prinsipal, atau 10 kondisi. Untuk informasi selengkapnya, lihat Kuota kebijakan Amazon SQS.

Kuota kebijakan Amazon SQS

Tabel berikut mencantumkan kuota yang terkait dengan kebijakan.

Nama	Maksimum
Byte	8,192
Ketentuan	10
Pengguna utama	50

Nama	Maksimum
Pernyataan	20
Tindakan per pernyataan	7

Fitur dan kemampuan Amazon SQS

Amazon SQS menyediakan fitur dan kemampuan berikut.

Topik

Menggunakan antrian huruf mati di Amazon SQS

Amazon SQS mendukung antrian huruf mati (DLQ), yang antrian sumber dapat menargetkan pesan yang tidak berhasil diproses. DLQ berguna untuk men-debug aplikasi Anda karena Anda dapat mengisolasi pesan yang tidak dikonsumsi untuk menentukan mengapa pemrosesan tidak berhasil. Untuk kinerja yang optimal, ini adalah praktik terbaik untuk menjaga antrian sumber dan DLQ dalam yang sama Akun AWS dan Wilayah. Setelah pesan berada dalam antrian surat mati, Anda dapat:

- Periksa log untuk pengecualian yang mungkin menyebabkan pesan dipindahkan ke antrian huruf mati.
- Menganalisis isi pesan yang dipindahkan ke antrian surat mati untuk mendiagnosis masalah aplikasi.
- Tentukan apakah Anda telah memberi konsumen Anda waktu yang cukup untuk memproses pesan.

- Pindahkan pesan dari antrian huruf mati menggunakan penggerak ulang antrian huruf mati.

Anda harus terlebih dahulu membuat antrian baru sebelum mengonfigurasinya sebagai antrian huruf mati. Untuk informasi tentang mengonfigurasi antrian huruf mati menggunakan konsol Amazon SQS, lihat. Pelajari cara mengonfigurasi antrian huruf mati menggunakan konsol Amazon SQS Untuk bantuan dengan antrian surat mati, seperti cara mengonfigurasi alarm untuk setiap pesan yang dipindahkan ke antrian huruf mati, lihat. Buat alarm untuk antrian huruf mati menggunakan Amazon CloudWatch

Menggunakan kebijakan untuk antrian surat mati

Gunakan kebijakan redrive untuk menentukan. `maxReceiveCount` `maxReceiveCount`Ini adalah berapa kali konsumen dapat menerima pesan dari antrian sumber sebelum dipindahkan ke antrian surat mati. Misalnya, jika `maxReceiveCount` disetel ke nilai rendah seperti 1, satu kegagalan untuk menerima pesan akan menyebabkan pesan pindah ke antrian huruf mati. Untuk memastikan bahwa sistem Anda tahan terhadap kesalahan, atur cukup `maxReceiveCount` tinggi untuk memungkinkan percobaan ulang yang cukup.

Kebijakan redrive allow menentukan antrian sumber mana yang dapat mengakses antrian huruf mati. Anda dapat memilih apakah akan mengizinkan semua antrian sumber, mengizinkan antrian sumber tertentu, atau menolak semua antrian sumber penggunaan antrian huruf mati. Default memungkinkan semua antrian sumber untuk menggunakan antrian huruf mati. Jika Anda memilih untuk mengizinkan antrian tertentu menggunakan `byQueue` opsi, Anda dapat menentukan hingga 10 antrian sumber menggunakan antrian sumber Amazon Resource Name (ARN). Jika Anda menentukan`denyAll`, antrian tidak dapat digunakan sebagai antrian huruf mati.

Memahami periode retensi pesan untuk antrian surat mati

Untuk antrian standar, kedaluwarsa pesan selalu didasarkan pada stempel waktu enqueue aslinya. Ketika pesan dipindahkan ke antrian huruf mati, stempel waktu enqueue tidak berubah. `ApproximateAgeOfOldestMessage`Metrik menunjukkan kapan pesan dipindahkan ke antrian huruf mati, bukan saat pesan awalnya dikirim. Misalnya, asumsikan bahwa pesan menghabiskan 1 hari dalam antrian asli sebelum dipindahkan ke antrian huruf mati. Jika periode retensi antrian surat mati adalah 4 hari, pesan akan dihapus dari antrian surat mati setelah 3 hari dan 3 hari. `ApproximateAgeOfOldestMessage` Dengan demikian, ini adalah praktik terbaik untuk selalu mengatur periode retensi antrian huruf mati menjadi lebih lama dari periode retensi antrian asli.

Untuk antrian FIFO, stempel waktu enqueue akan disetel ulang saat pesan dipindahkan ke antrian huruf mati. `ApproximateAgeOfOldestMessage`Metrik menunjukkan kapan pesan dipindahkan ke antrian huruf mati. Dalam contoh yang sama di atas, pesan dihapus dari antrian huruf mati setelah 4 hari dan `ApproximateAgeOfOldestMessage` adalah 4 hari.

Pelajari cara mengonfigurasi antrian huruf mati menggunakan konsol Amazon SQS

Antrian huruf mati adalah antrian yang dapat ditargetkan antrian sumber untuk pesan yang tidak berhasil diproses. Untuk informasi selengkapnya, lihat Menggunakan antrian huruf mati di Amazon SQS .

Amazon SQS tidak membuat antrian huruf mati secara otomatis. Anda harus terlebih dahulu membuat antrian sebelum menggunakannya sebagai antrian huruf mati. Untuk petunjuk tentang membuat antrian untuk digunakan sebagai antrian huruf mati, lihat. Buat antrian menggunakan konsol Amazon SQS

Antrean dead-letter pada antrean FIFO juga harus merupakan antrean FIFO. Demikian pula, antrean dead-letter dari antrean standar juga harus menjadi antrean standar.

Saat membuat atau mengedit antrian, Anda dapat mengonfigurasi antrian huruf mati.

Untuk mengonfigurasi antrian huruf mati untuk antrian yang ada (konsol)

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian.
3. Pilih antrian dan pilih Edit.
4. Gulir ke bagian antrian Dead-letter dan pilih Diaktifkan.
5. Pilih Nama Sumber Daya Amazon (ARN) dari Antrian Surat Mati yang ada yang ingin Anda kaitkan dengan antrian sumber ini.
6. Untuk mengonfigurasi berapa kali pesan dapat diterima sebelum dikirim ke antrian huruf mati, atur Maksimum menerima ke nilai antara 1 dan 1.000.
7. Setelah Anda selesai mengonfigurasi antrian huruf mati, pilih Simpan.

   Setelah Anda menyimpan antrian, konsol akan menampilkan halaman Detail untuk antrian Anda. Pada halaman Detail, tab antrian Dead-letter menampilkan ARN Penerima Maksimum dan Antrian Surat Mati dalam antrian Dead-letter.

Pelajari cara mengonfigurasi redrive antrian huruf mati di Amazon SQS

Anda dapat menggunakan penggerak ulang antrian huruf mati untuk memindahkan pesan yang tidak dikonsumsi dari antrian huruf mati yang ada. Secara default, redrive antrian huruf mati memindahkan pesan dari antrian huruf mati ke antrian sumber. Namun, Anda juga dapat mengonfigurasi antrian lain sebagai tujuan redrive jika kedua antrian adalah tipe yang sama. Misalnya, jika antrian huruf mati adalah antrian FIFO, antrian tujuan redrive harus berupa antrian FIFO juga. Selain itu, Anda dapat mengonfigurasi kecepatan redrive untuk mengatur kecepatan di mana Amazon SQS memindahkan pesan.

> **Note**
>
> Ketika pesan dipindahkan dari antrian FIFO ke DLQ FIFO, ID deduplikasi pesan asli akan diganti dengan ID pesan asli. Ini untuk memastikan bahwa deduplikasi DLQ tidak akan mencegah penyimpanan dua pesan independen yang kebetulan berbagi ID deduplikasi.

Antrian surat mati menggerakkan ulang pesan sesuai urutan penerimaannya, dimulai dengan pesan tertua. Namun, antrian tujuan menyerap pesan yang digerakkan ulang, serta pesan baru dari produsen lain, sesuai dengan urutan penerimaannya. Misalnya, jika produser mengirim pesan ke antrian FIFO sumber saat secara bersamaan menerima pesan yang digerakkan ulang dari antrian surat mati, pesan yang digerakkan ulang akan terjalin dengan pesan baru dari produser.

> **Note**
>
> Tugas redrive mengatur ulang periode retensi. Semua pesan yang digerakkan ulang dianggap pesan baru dengan pesan baru `messageID` dan ditetapkan ke pesan `enqueueTime` yang digerakkan ulang.

Topik

- Mengonfigurasi redrive antrian huruf mati untuk antrean standar yang ada menggunakan Amazon SQS API
- Mengonfigurasi redrive antrian huruf mati untuk antrean standar yang ada menggunakan konsol Amazon SQS
- Mengkonfigurasi izin antrian untuk redrive antrian huruf mati

Mengonfigurasi redrive antrian huruf mati untuk antrean standar yang ada menggunakan Amazon SQS API

Anda dapat mengonfigurasi redrive antrian huruf mati menggunakan tindakan`SendMessageBatch`,`ReceiveMessage`, dan API: `DeleteMessageBatch`

Tindakan API	Deskripsi
`StartMessageMoveTask`	Memulai tugas asinkron untuk memindahkan pesan dari antrian sumber tertentu ke antrian tujuan tertentu.
`ListMessageMoveTasks`	Mendapat tugas pergerakan pesan terbaru (hingga 10) di bawah antrian sumber tertentu.
`CancelMessageMoveTask`	Membatalkan tugas pergerakan pesan tertentu. Pergerakan pesan hanya dapat dibatalkan ketika status saat ini sedang BERJALAN.

Mengonfigurasi redrive antrian huruf mati untuk antrean standar yang ada menggunakan konsol Amazon SQS

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian.
3. Pilih nama antrian yang telah Anda konfigurasikan sebagai antrian huruf mati.
4. Pilih Mulai DLQ redrive.
5. Di bawah konfigurasi Recrive, untuk tujuan Pesan, lakukan salah satu hal berikut:
   - Untuk mengarahkan ulang pesan ke antrean sumbernya, pilih Recrive to source queue (s).
   - Untuk mengarahkan ulang pesan ke antrian lain, pilih Recrive ke tujuan kustom. Kemudian, masukkan Nama Sumber Daya Amazon (ARN) dari antrian tujuan yang ada.
6. Di bawah Pengaturan kontrol kecepatan, pilih salah satu dari berikut ini:
   - Sistem dioptimalkan - Dorong ulang pesan antrian huruf mati dengan jumlah maksimum pesan per detik.

- Kecepatan maks khusus - Dorong ulang pesan antrian huruf mati dengan kecepatan maksimum pesan per detik khusus. Tarif maksimum yang diizinkan adalah 500 pesan per detik.
  - Disarankan untuk memulai dengan nilai kecil untuk kecepatan maks Kustom dan memverifikasi bahwa antrian sumber tidak kewalahan dengan pesan. Dari sana, secara bertahap tingkatkan nilai kecepatan maks Kustom, terus memantau status antrian sumber.

7. Setelah Anda selesai mengonfigurasi redrive antrian huruf mati, pilih Recrive messages.

> **Important**
>
> Amazon SQS tidak mendukung pemfilteran dan modifikasi pesan saat mengarahkan ulang pesan dari antrian huruf mati.
> Tugas redrive antrian huruf mati dapat berjalan maksimal 36 jam. Amazon SQS mendukung maksimal 100 tugas redrive aktif per akun.

8. Jika Anda ingin membatalkan tugas redrive pesan, pada halaman Detail untuk antrian Anda, pilih Batalkan drive ulang DLQ. Saat membatalkan redrive pesan yang sedang berlangsung, pesan apa pun yang telah berhasil dipindahkan ke antrian tujuan pindahnya akan tetap berada dalam antrean tujuan.

Mengkonfigurasi izin antrian untuk redrive antrian huruf mati

Anda dapat memberi pengguna akses ke tindakan antrean huruf mati tertentu dengan menambahkan izin ke kebijakan Anda. Izin minimum yang diperlukan untuk redrive antrian huruf mati adalah sebagai berikut:

Izin Minimum	Metode API yang diperlukan
Untuk memulai redrive pesan	• Tambahkan`sqs:StartMessageMoveTask` ,, `sqs:ReceiveMessage` `sqs:DeleteMessage` , dan `sqs:GetQueueAttributes` antrian huruf mati. Jika antrian huruf mati atau antrian sumber asli dienkripsi (juga dikenal sebagai antrian SSE), `kms:Decrypt` untuk kunci KMS apa pun yang telah digunakan untuk mengenkripsi pesan juga diperlukan. •

Izin Minimum	Metode API yang diperlukan
	Tambahkan antrian tujuan. `sqs:SendMessage` Jika antrian tujuan dienkripsi, `kms:GenerateDataKey` dan `kms:Decrypt` juga diperlukan.
Untuk membatalkan redrive pesan yang sedang berlangsung	• Tambahkan`sqs:CancelMessageMoveTask` ,, `sqs:ReceiveMessage` `sqs:DeleteMessage` , dan `sqs:GetQueueAttributes` antrian huruf mati. Jika antrian huruf mati dienkripsi (juga dikenal sebagai antrian SSE), juga diperlukan. `kms:Decrypt`
Untuk menampilkan status pemindahan pesan	• Tambahkan `sqs:ListMessageMoveTasks` dan `sqs:GetQueueAttributes` antrian huruf mati.

Untuk mengonfigurasi izin untuk pasangan antrian terenkripsi (antrian sumber dengan antrian huruf mati)

Gunakan langkah-langkah berikut untuk mengonfigurasi izin minimum untuk redrive antrian huruf mati:

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Buat kebijakan dengan izin berikut dan lampirkan ke pengguna atau peran IAM login Anda:
   - `sqs:StartMessageMoveTask`
   - `sqs:CancelMessageMoveTask`
   - `sqs:ListMessageMoveTasks`
   - `sqs:ListDeadLetterSourceQueues`
   - `sqs:ReceiveMessage`
   - `sqs:DeleteMessage`
   - `sqs:GetQueueAttributes`

- `Resource`<DLQ_region><DLQ_accountId><DLQ_name>ARN dari antrian huruf mati (misalnya, “arn:aws:sqs::: “).
- `sqs:SendMessage`
- `Resource`ARN dari antrian tujuan (misalnya, “arn:aws:sqs: *< DestQueue _region>: < _accountID>: < _name>* “). DestQueue DestQueue
- `kms:Decrypt`— Memungkinkan tindakan dekripsi.
- `kms:GenerateDataKey`
- `Resource`<region><accountId><keyId_used to encrypt the message body>ARN dari kunci enkripsi KMS apa pun yang telah digunakan untuk mengenkripsi pesan dalam antrian sumber asli (misalnya, “arn:aws:kms: ::key/ “).
- <region><accountId><keyId_used for the destination queue>ARN Sumber Daya dari kunci enkripsi KMS yang digunakan untuk antrian tujuan redrive (misalnya, “arn:aws:kms: ::key/ “).

Kebijakan akses Anda harus menyerupai yang berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sqs:StartMessageMoveTask",
        "sqs:CancelMessageMoveTask",
        "sqs:ListMessageMoveTasks",
        "sqs:ReceiveMessage",
        "sqs:DeleteMessage",
        "sqs:GetQueueAttributes",
        "sqs:ListDeadLetterSourceQueues"
      ],
      "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>"
    }
  ]
}
```

Untuk mengonfigurasi izin menggunakan pasangan antrian yang tidak terenkripsi (antrian sumber dengan antrian huruf mati)

Gunakan langkah-langkah berikut untuk mengonfigurasi izin minimum untuk antrian huruf mati standar yang tidak terenkripsi. Izin minimum yang diperlukan adalah menerima, menghapus, dan mendapatkan atribut dari antrian huruf mati, dan mengirim atribut ke antrian sumber.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Buat kebijakan dengan izin berikut dan lampirkan ke pengguna atau peran IAM login Anda:

   - `sqs:StartMessageMoveTask`
   - `sqs:CancelMessageMoveTask`
   - `sqs:ListMessageMoveTasks`
   - `sqs:ListDeadLetterSourceQueues`
   - `sqs:ReceiveMessage`
   - `sqs:DeleteMessage`
   - `sqs:GetQueueAttributes`
   - `Resource`<DLQ_region><DLQ_accountId><DLQ_name>ARN dari antrian huruf mati (misalnya, “arn:aws:sqs::: “).
   - `sqs:SendMessage`
   - `Resource`*ARN dari antrian tujuan (misalnya, “arn:aws:sqs: < DestQueue _region>: < _accountID>: < _name> “). DestQueue DestQueue*

   Kebijakan akses Anda harus menyerupai yang berikut:

```
{
"Version": "2012-10-17",
  "Statement": [
    {
"Effect": "Allow",
      "Action": [
        "sqs:StartMessageMoveTask",
        "sqs:CancelMessageMoveTask",
        "sqs:ListMessageMoveTasks",
        "sqs:ReceiveMessage",
        "sqs:DeleteMessage",
        "sqs:GetQueueAttributes",
        "sqs:ListDeadLetterSourceQueues"
      ],
      "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>"
    }
  ]
}
```

CloudTrail persyaratan pembaruan dan izin untuk redrive antrian surat mati Amazon SQS

Pada 8 Juni 2023, Amazon SQS memperkenalkan redrive antrian huruf mati (DLQ) untuk SDK dan (CLI). AWS AWS Command Line Interface Kemampuan ini merupakan tambahan untuk redrive DLQ yang sudah didukung untuk konsol. AWS Jika sebelumnya Anda telah menggunakan AWS konsol untuk mengarahkan ulang pesan antrian huruf mati, Anda mungkin terpengaruh oleh perubahan berikut:

- CloudTrail penggantian nama acara untuk redrive antrian huruf mati
- Izin yang diperbarui untuk redrive antrian huruf mati

CloudTrail penggantian nama acara

Pada 15 Oktober 2023, nama CloudTrail acara untuk redrive antrian huruf mati akan berubah di konsol Amazon SQS. Jika Anda telah menyetel alarm untuk CloudTrail acara ini, Anda harus memperbaruinya sekarang. Berikut ini adalah nama CloudTrail acara baru untuk DLQ redrive:

Nama acara sebelumnya	Nama acara baru
`CreateMoveTask`	`StartMessageMoveTask`
`CancelMoveTask`	`CancelMessageMoveTask`

Izin yang diperbarui

Termasuk dengan rilis SDK dan CLI, Amazon SQS juga telah memperbarui izin antrian untuk redrive DLQ untuk mematuhi praktik terbaik keamanan. Gunakan jenis izin antrian berikut untuk mengarahkan ulang pesan dari DLQ Anda.

1. Izin berbasis tindakan (pembaruan untuk tindakan API DLQ)
2. Izin kebijakan Amazon SQS terkelola
3. Kebijakan izin yang menggunakan sqs:* wildcard

> **Important**
>
> Untuk menggunakan redrive DLQ untuk SDK atau CLI, Anda harus memiliki kebijakan izin redrive DLQ yang cocok dengan salah satu opsi di atas.

Jika izin antrian Anda untuk drive ulang DLQ tidak cocok dengan salah satu opsi di atas, Anda harus memperbarui izin Anda sebelum 31 Agustus 2023. Antara sekarang dan 31 Agustus 2023, akun Anda akan dapat menggerakkan ulang pesan menggunakan izin yang Anda konfigurasikan menggunakan AWS konsol hanya di wilayah tempat Anda sebelumnya menggunakan drive ulang DLQ. Misalnya, Anda memiliki “Akun A” di us-east-1 dan eu-west-1. “Akun A” digunakan untuk menggerakkan ulang pesan di AWS konsol di us-east-1 sebelum 8 Juni 2023, tetapi tidak di eu-west-1. Antara 8 Juni 2023 dan 31 Agustus 2023, jika izin kebijakan “Akun A” tidak cocok dengan salah satu opsi di atas, itu hanya dapat digunakan untuk menggerakkan ulang pesan di konsol AWS di us-east-1, dan bukan di eu-west-1.

> **Important**
>
> Jika izin redrive DLQ Anda tidak cocok dengan salah satu opsi ini setelah 31 Agustus 2023, akun Anda tidak akan lagi dapat menggerakkan ulang pesan DLQ menggunakan konsol. AWS
> Namun, jika Anda menggunakan fitur redrive DLQ di AWS Konsol selama Agustus 2023, Anda memiliki ekstensi hingga 15 Oktober 2023 untuk mengadopsi izin baru sesuai dengan salah satu opsi ini.
> Untuk informasi selengkapnya, lihat the section called “Mengidentifikasi kebijakan yang terkena dampak”.

Berikut ini adalah contoh izin antrian untuk setiap opsi redrive DLQ. Saat menggunakan antrian terenkripsi sisi server (SSE), izin kunci yang sesuai diperlukan. AWS KMS

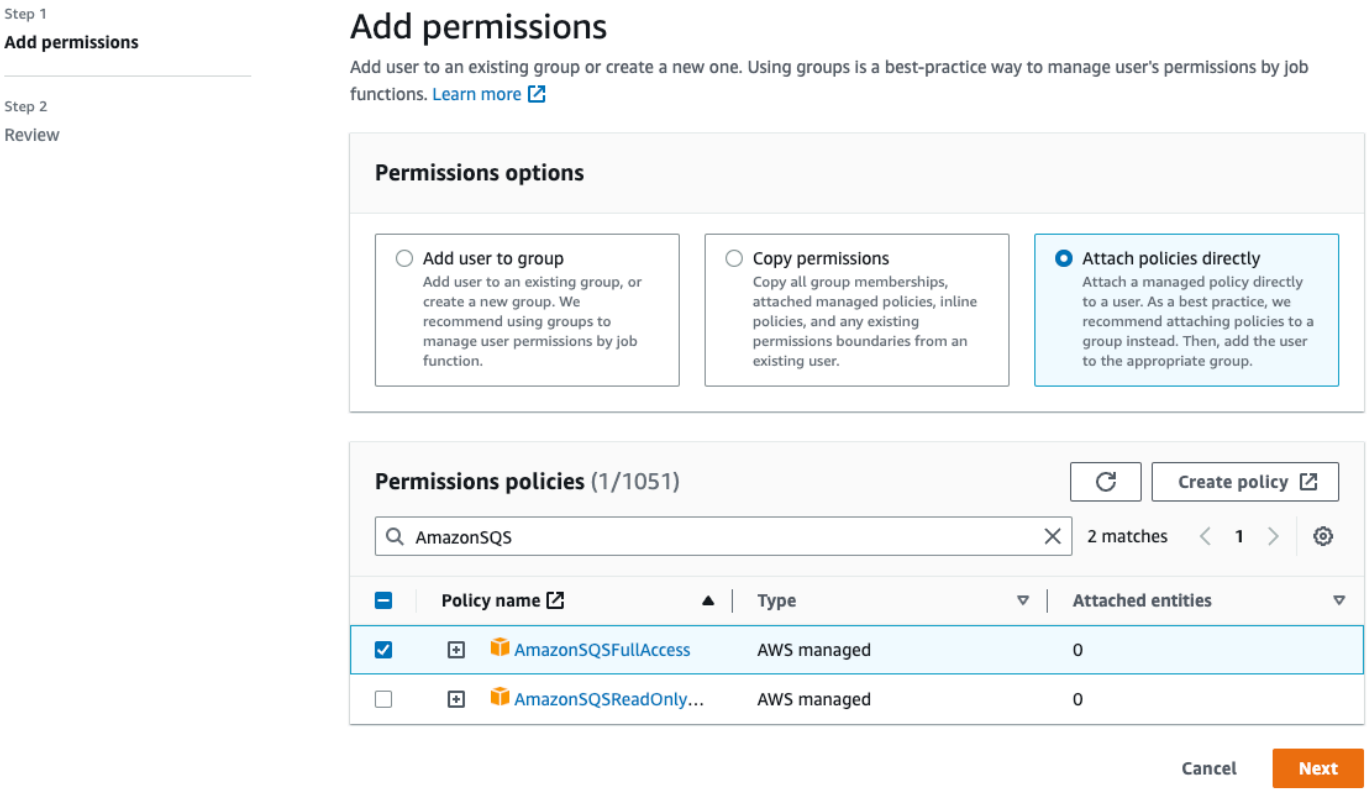
Berbasis aksi

```
{
  "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>"
    }
  ]
}
```

Kebijakan terkelola

Kebijakan terkelola berikut berisi izin diperbarui yang diperlukan:

- AmazonSQS FullAccess - Termasuk tugas redrive antrian huruf mati berikut: mulai, batalkan, dan daftar.
- Akses AmazonSQS - Menyediakan ReadOnly akses hanya-baca, dan menyertakan daftar tugas redrive antrian huruf mati.



Kebijakan Izin yang menggunakan sqs* wildcard

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

```
}
```

Mengidentifikasi kebijakan yang terkena dampak

Jika Anda menggunakan kebijakan terkelola pelanggan (CMP), Anda dapat menggunakan AWS CloudTrail dan IAM untuk mengidentifikasi kebijakan yang dipengaruhi oleh pembaruan izin antrian.

> **Note**
>
> Jika Anda menggunakan `AmazonSQSFullAccess` dan`AmazonSQSReadOnlyAccess`, tidak ada tindakan lebih lanjut yang diperlukan.

1. Masuk ke AWS CloudTrail konsol.
2. Pada halaman Riwayat acara, di bawah Cari atribut, gunakan menu tarik-turun untuk memilih nama Acara. Kemudian, cari`CreateMoveTask`.
3. Pilih acara untuk membuka halaman Detail. Di bagian Catatan peristiwa, ambil `UserName` atau `RoleName` dari `userIdentity` ARN.
4. Masuk ke konsol IAM.
   - Untuk pengguna, pilih Pengguna. Pilih pengguna dengan yang `UserName` diidentifikasi pada langkah sebelumnya.
   - Untuk peran, pilih Peran. Cari pengguna dengan yang `RoleName` diidentifikasi pada langkah sebelumnya.
5. Pada halaman Detail, di bagian Izin, tinjau kebijakan apa pun dengan `sqs:` awalan di`Action`, atau tinjau kebijakan yang menentukan antrean Amazon SQS. `Resource`

Buat alarm untuk antrian huruf mati menggunakan Amazon CloudWatch

Anda dapat mengonfigurasi alarm untuk setiap pesan yang dipindahkan ke antrian huruf mati menggunakan Amazon CloudWatch dan metrik. `ApproximateNumberOfMessagesVisible` Untuk informasi selengkapnya, lihat Membuat CloudWatch alarm untuk metrik Amazon SQS. Setelah menerima peringatan bahwa pesan telah dikirim ke antrian surat mati, Anda dapat meninjau pesan menggunakan polling untuk menerima pesan.

Metadata pesan untuk Amazon SQS

Anda dapat menggunakan atribut pesan untuk melampirkan metadata kustom ke pesan Amazon SQS untuk aplikasi Anda. Anda dapat menggunakan atribut sistem pesan untuk menyimpan metadata untuk AWS layanan lain, seperti. AWS X-Ray

Topik

Atribut pesan Amazon SQS

Amazon SQS memungkinkan Anda menyertakan metadata terstruktur (seperti stempel waktu, data geospasial, tanda tangan, dan pengidentifikasi) dengan pesan yang menggunakan atribut pesan. Setiap pesan dapat memiliki hingga 10 atribut. Atribut pesan bersifat opsional dan terpisah dari badan pesan (namun, atribut tersebut dikirim di sampingnya). Konsumen Anda dapat menggunakan atribut pesan untuk menangani pesan dengan cara tertentu tanpa harus memproses isi pesan terlebih dahulu. Untuk informasi tentang mengirim pesan dengan atribut menggunakan konsol Amazon SQS, lihat. Mengirim pesan dengan atribut

> **Note**
>
> Jangan bingung atribut pesan dengan atribut sistem pesan: Meskipun Anda dapat menggunakan atribut pesan untuk melampirkan metadata kustom ke pesan Amazon SQS untuk aplikasi Anda, Anda dapat menggunakan atribut sistem pesan untuk menyimpan metadata untuk layanan lain, seperti. AWS AWS X-Ray

Topik

Komponen atribut pesan

> **Important**
>
> Semua komponen atribut pesan disertakan dalam pembatasan ukuran pesan 256 KB. Badan pesan `Name` `TypeValue`,,, dan pesan tidak boleh kosong atau nol.

Setiap atribut pesan terdiri dari komponen-komponen berikut:

- Nama - Nama atribut pesan dapat berisi karakter berikut: `A` -`Z`, `-`, a `0` - `z9`, garis bawah (_), tanda hubung (), dan periode (`-`). `.` Pembatasan berikut berlaku:
  - Panjangnya bisa sampai 256 karakter
  - Tidak dapat memulai dengan `AWS.` atau `Amazon.` (atau variasi casing apa pun)
  - Peka huruf besar/kecil
  - Harus unik di antara semua nama atribut untuk pesan
  - Tidak boleh dimulai atau diakhiri dengan titik
  - Tidak boleh memiliki periode secara berurutan
- Jenis - Jenis data atribut pesan. Jenis yang didukung meliputi`String`,`Number`, dan`Binary`. Anda juga dapat menambahkan informasi khusus untuk tipe data apa pun. Tipe data memiliki batasan yang sama dengan isi pesan (untuk informasi selengkapnya, lihat `SendMessage` di Referensi API Layanan Antrian Sederhana Amazon). Selain itu, pembatasan berikut berlaku:
  - Panjangnya bisa sampai 256 karakter
  - Peka huruf besar/kecil
- Nilai - Nilai atribut pesan. Untuk tipe `String` data, nilai atribut memiliki batasan yang sama dengan isi pesan.

Tipe data atribut pesan

Tipe data atribut pesan menginstruksikan Amazon SQS cara menangani nilai atribut pesan yang sesuai. Misalnya, jika jenisnya`Number`, Amazon SQS memvalidasi nilai numerik.

Amazon SQS mendukung tipe data logis `String``Number`, dan `Binary` dengan label tipe data kustom opsional dengan format *`.custom-data-type`*

- String - `String` atribut dapat menyimpan teks Unicode menggunakan karakter XMLyang valid.

- Angka — `Number` atribut dapat menyimpan nilai numerik positif atau negatif. Sebuah angka dapat memiliki hingga 38 digit presisi, dan bisa antara 10^-128 dan 10^+126.

  > **Note**
  >
  > Amazon SQS menghapus angka nol terdepan dan tertinggal.

- Biner — Atribut biner dapat menyimpan data biner apa pun seperti data terkompresi, data terenkripsi, atau gambar.
- Kustom - Untuk membuat tipe data kustom, tambahkan label tipe khusus ke tipe data apa pun. Sebagai contoh:
  - `Number.byte`,`Number.short`,`Number.int`, dan `Number.float` dapat membantu membedakan antara jenis angka.
  - `Binary.gif`dan `Binary.png` dapat membantu membedakan antara jenis file.

  > **Note**
  >
  > Amazon SQS tidak menafsirkan, memvalidasi, atau menggunakan data yang ditambahkan. Label tipe khusus memiliki batasan yang sama dengan isi pesan.

Menghitung intisari pesan MD5 untuk atribut pesan

Jika Anda menggunakan AWS SDK for Java, Anda dapat melewati bagian ini. `MessageMD5ChecksumHandler`Kelas SDK for Java mendukung intisari pesan MD5 untuk atribut pesan Amazon SQS.

Jika Anda menggunakan API Kueri atau salah satu AWS SDK yang tidak mendukung intisari pesan MD5 untuk atribut pesan Amazon SQS, Anda harus menggunakan panduan berikut untuk melakukan perhitungan intisari pesan MD5.

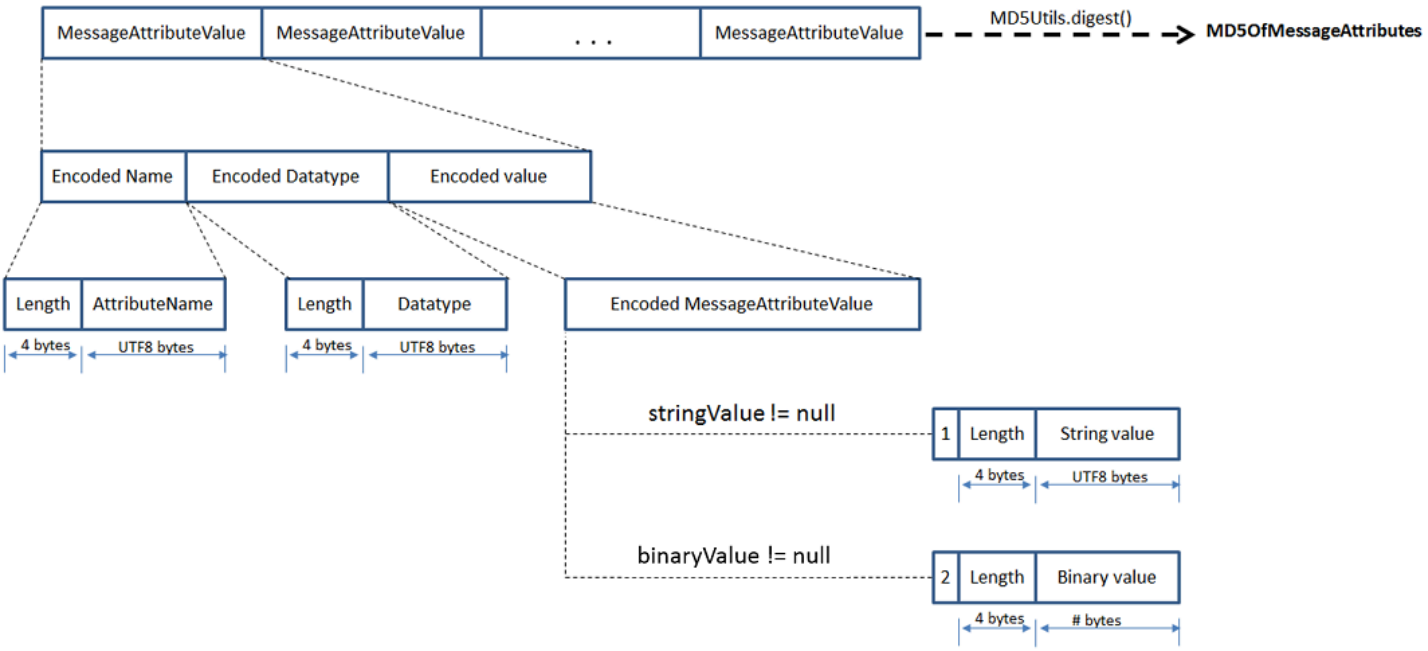
> **Note**
>
> Selalu sertakan sufiks tipe data kustom dalam perhitungan intisari pesan MD5.

Gambaran Umum

Berikut ini adalah ikhtisar algoritma perhitungan intisari pesan MD5:

1. Urutkan semua atribut pesan berdasarkan nama dalam urutan menaik.
2. Encode masing-masing bagian dari setiap atribut (`Name`,`Type`, dan`Value`) ke dalam buffer.
3. Hitung intisari pesan dari seluruh buffer.

Diagram berikut menunjukkan pengkodean intisari pesan MD5 untuk atribut pesan tunggal:



Untuk menyandikan satu atribut pesan Amazon SQS

1. Encode nama: panjang (4 byte) dan UTF-8 byte dari nama.
2. Encode tipe data: panjang (4 byte) dan UTF-8 byte dari tipe data.
3. Mengkodekan jenis transport (`String`atau`Binary`) dari nilai (1 byte).

> **Note**
>
> Tipe data logis `String` dan `Number` menggunakan tipe `String` transport.
> Tipe data logis `Binary` menggunakan tipe `Binary` transport.

   a. Untuk jenis `String` transportasi, encode 1.

   b. Untuk jenis `Binary` transportasi, encode 2.

4. Mengkodekan nilai atribut.

   a. Untuk tipe `String` transport, encode nilai atribut: panjang (4 byte) dan UTF-8 byte dari nilai.

   b. Untuk tipe `Binary` transport, encode nilai atribut: panjang (4 byte) dan byte mentah dari nilai.

Atribut sistem pesan Amazon SQS

Meskipun Anda dapat menggunakan atribut pesan untuk melampirkan metadata kustom ke pesan Amazon SQS untuk aplikasi Anda, Anda dapat menggunakan atribut sistem pesan untuk menyimpan metadata untuk AWS layanan lain, seperti. AWS X-Ray Untuk informasi selengkapnya, lihat parameter `MessageSystemAttribute` permintaan tindakan `SendMessage` dan `SendMessageBatch` API, `AWSTraceHeader` atribut tindakan `ReceiveMessage` API, dan tipe `MessageSystemAttributeValue` data di Referensi API Layanan Antrian Sederhana Amazon.

Atribut sistem pesan terstruktur persis seperti atribut pesan, dengan pengecualian berikut:

- Saat ini, satu-satunya atribut sistem pesan yang didukung adalah`AWSTraceHeader`. Tipenya harus `String` dan nilainya harus berupa string header AWS X-Ray jejak yang diformat dengan benar.
- Ukuran atribut sistem pesan tidak dihitung terhadap ukuran total pesan.

Sumber daya yang diperlukan untuk memproses pesan Amazon SQS

Untuk membantu memperkirakan sumber daya yang Anda perlukan untuk memproses pesan antrian, Amazon SQS dapat menentukan perkiraan jumlah pesan yang tertunda, terlihat, dan tidak terlihat dalam antrian. Untuk informasi selengkapnya tentang visibilitas, lihatBatas waktu visibilitas Amazon SQS.

> **Note**
>
> Untuk antrian standar, hasilnya adalah perkiraan karena arsitektur terdistribusi Amazon SQS. Dalam kebanyakan kasus, hitungan harus mendekati jumlah sebenarnya dari pesan dalam antrian.

Untuk antrian FIFO, hasilnya tepat.

Tabel berikut mencantumkan nama atribut yang akan digunakan dengan `GetQueueAttributes` tindakan:

Tugas	Nama atribut
Dapatkan perkiraan jumlah pesan yang tersedia untuk diambil dari antrian.	`ApproximateNumberOfMessages Visible`
Dapatkan perkiraan jumlah pesan dalam antrian yang tertunda dan tidak tersedia untuk dibaca segera. Hal ini dapat terjadi ketika antrian dikonfigurasi sebagai antrian penundaan atau ketika pesan telah dikirim dengan parameter delay.	`ApproximateNumberOfMessages Delayed`
Dapatkan perkiraan jumlah pesan yang sedang dalam penerbangan. Pesan dianggap dalam penerbangan jika telah dikirim ke klien tetapi belum dihapus atau belum mencapai akhir jendela visibilitas mereka.	`ApproximateNumberOfMessages NotVisible`

Daftar antrian pagination

Metode `listQueues` dan `listDeadLetterQueues` API mendukung kontrol pagination opsional. Secara default, metode API ini mengembalikan hingga 1000 antrian dalam pesan respons. Anda dapat mengatur `MaxResults` parameter untuk mengembalikan hasil yang lebih sedikit di setiap respons.

Tetapkan parameter `MaxResults` dalam `listDeadLetterQueues`permintaan `listQueues`atau untuk menentukan jumlah maksimum hasil yang akan dikembalikan dalam respons. Jika Anda tidak mengatur`MaxResults`, respon mencakup maksimal 1.000 hasil dan `NextToken` nilai dalam respon adalah nol.

Jika Anda mengaturMaxResults, respons menyertakan nilai untuk NextToken jika ada hasil tambahan untuk ditampilkan. Gunakan NextToken sebagai parameter dalam permintaan Anda berikutnya listQueues untuk menerima halaman hasil berikutnya. Jika tidak ada hasil tambahan untuk ditampilkan, NextToken nilai dalam respons adalah nol.

Tag alokasi biaya Amazon SQS

Untuk mengatur dan mengidentifikasi antrian Amazon SQS untuk alokasi biaya, Anda dapat menambahkan tag metadata yang mengidentifikasi tujuan, pemilik, atau lingkungan antrian. Ini sangat berguna ketika Anda memiliki banyak antrian. Untuk mengonfigurasi tag menggunakan konsol Amazon SQS, lihat the section called “Mengkonfigurasi tag untuk antrian”

Anda dapat menggunakan tag alokasi biaya untuk mengatur AWS tagihan Anda untuk mencerminkan struktur biaya Anda sendiri. Untuk melakukan ini, daftar untuk mendapatkan Akun AWS tagihan Anda untuk menyertakan kunci tag dan nilai. Untuk informasi selengkapnya, lihat Menyiapkan Laporan Alokasi Biaya Bulanan di Panduan Pengguna AWS Billing .

Setiap tag terdiri dari pasangan kunci-nilai yang Anda tentukan. Misalnya, Anda dapat dengan mudah mengidentifikasi antrian produksi dan pengujian jika Anda menandai antrian Anda sebagai berikut:

Antrean	Kunci	Nilai
MyQueueA	QueueType	Production
MyQueueB	QueueType	Testing

> **Note**
>
> Saat Anda menggunakan tag antrian, ingatlah pedoman berikut:
>
> - Kami tidak menyarankan menambahkan lebih dari 50 tag ke antrian. Penandaan mendukung karakter Unicode di UTF-8.
> - Tag tidak memiliki arti semantik. Amazon SQS menafsirkan tag sebagai string karakter.
> - Tag peka huruf besar/kecil.
> - Tag baru dengan kunci yang identik dengan tag yang ada menimpa tag yang ada.
> - Tindakan penandaan dibatasi hingga 30 TPS per. Akun AWS Jika aplikasi Anda membutuhkan throughput yang lebih tinggi, kirimkan permintaan.

Untuk daftar lengkap pembatasan tag, lihatKuota.

Amazon SQS polling pendek dan panjang

Amazon SQS menawarkan opsi polling pendek dan panjang untuk menerima pesan dari antrian. Pertimbangkan persyaratan aplikasi Anda untuk responsif dan efisiensi biaya saat memilih di antara dua opsi pemungutan suara ini:

- Jajak pendapat singkat (default) — `ReceiveMessage`Permintaan meminta subset server (berdasarkan distribusi acak tertimbang) untuk menemukan pesan yang tersedia dan mengirimkan respons langsung, bahkan jika tidak ada pesan yang ditemukan.
- Polling panjang — **`ReceiveMessage`**menanyakan semua server untuk pesan, mengirim respons sekali setidaknya satu pesan tersedia, hingga maksimum yang ditentukan. Respons kosong dikirim hanya jika waktu tunggu polling berakhir. Opsi ini dapat mengurangi jumlah tanggapan kosong dan biaya yang berpotensi lebih rendah.

Bagian berikut menjelaskan rincian polling pendek dan polling panjang.

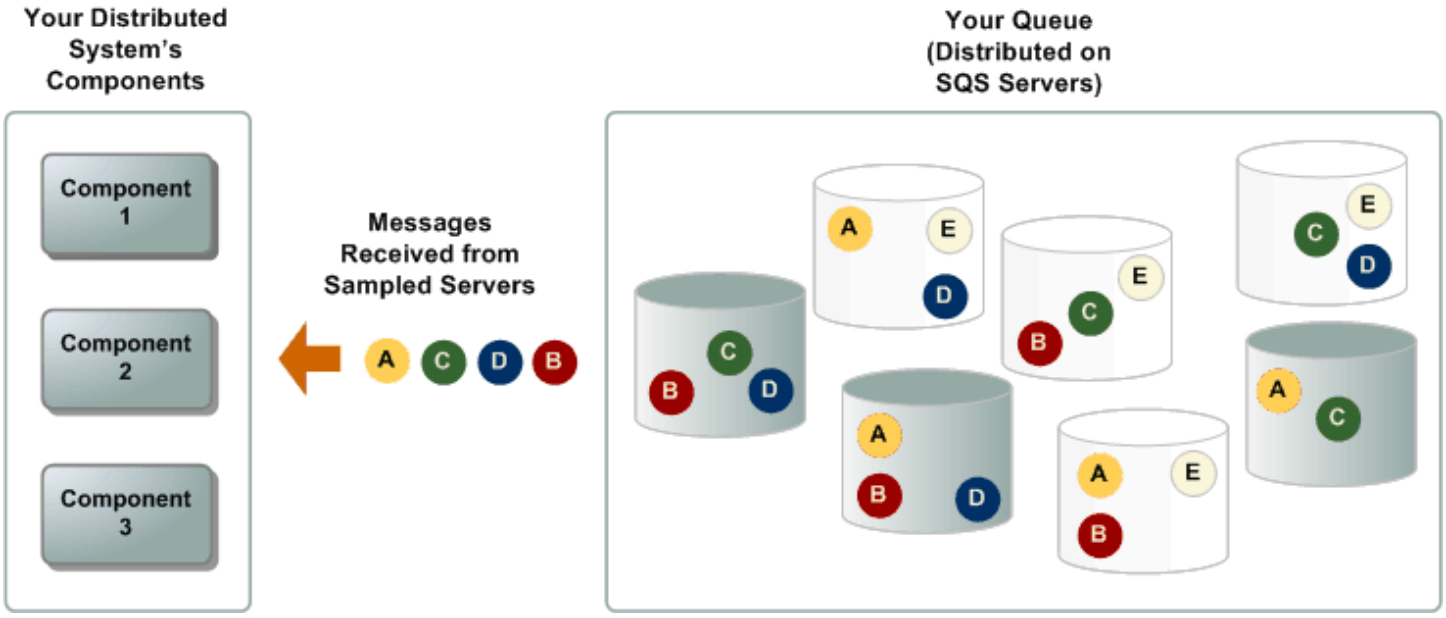
Topik

- Mengkonsumsi pesan menggunakan polling singkat
- Mengkonsumsi pesan menggunakan polling panjang
- Perbedaan antara polling panjang dan pendek

Mengkonsumsi pesan menggunakan polling singkat

Saat Anda menggunakan pesan dari antrian (FIFO atau standar) menggunakan polling singkat, Amazon SQS mengambil sampel subset servernya (berdasarkan distribusi acak tertimbang) dan menampilkan pesan hanya dari server tersebut. Dengan demikian, `ReceiveMessage`permintaan tertentu mungkin tidak mengembalikan semua pesan Anda. Namun, jika Anda memiliki kurang dari 1.000 pesan dalam antrian Anda, permintaan berikutnya akan mengembalikan pesan Anda. Jika Anda terus mengkonsumsi dari antrian Anda, Amazon SQS mengambil sampel semua servernya, dan Anda menerima semua pesan Anda.

Diagram berikut menunjukkan perilaku polling singkat pesan yang dikembalikan dari antrian standar setelah salah satu komponen sistem Anda membuat permintaan terima. Amazon SQS mengambil sampel beberapa servernya (berwarna abu-abu) dan mengembalikan pesan A, C, D, dan B dari server ini. Pesan E tidak dikembalikan untuk permintaan ini, tetapi dikembalikan untuk permintaan berikutnya.



Mengkonsumsi pesan menggunakan polling panjang

Ketika waktu tunggu untuk tindakan `ReceiveMessage` API lebih besar dari 0, polling panjang berlaku. Waktu tunggu polling maksimum yang panjang adalah 20 detik. Polling panjang membantu mengurangi biaya penggunaan Amazon SQS dengan menghilangkan jumlah respons kosong (bila tidak ada pesan yang tersedia untuk `ReceiveMessage` permintaan) dan respons kosong palsu (saat pesan tersedia tetapi tidak disertakan dalam respons). Untuk informasi tentang mengaktifkan polling panjang untuk antrean baru atau yang sudah ada menggunakan konsol Amazon SQS, lihat. Mengkonfigurasi parameter antrian menggunakan konsol Amazon SQS Untuk praktik terbaik, lihat Menyiapkan polling panjang.

Jajak pendapat panjang menawarkan manfaat berikut:

- Kurangi respons kosong dengan mengizinkan Amazon SQS menunggu hingga pesan tersedia dalam antrian sebelum mengirim respons. Kecuali waktu koneksi habis, respons terhadap `ReceiveMessage` permintaan berisi setidaknya satu pesan yang tersedia, hingga jumlah maksimum pesan yang ditentukan dalam `ReceiveMessage` tindakan. Dalam kasus yang jarang terjadi, Anda mungkin menerima respons kosong bahkan

ketika antrian masih berisi pesan, terutama jika Anda menentukan nilai rendah untuk `ReceiveMessageWaitTimeSeconds`parameter tersebut.

- Kurangi respons kosong palsu dengan menanyakan semua—bukan subset dari—server Amazon SQS.
- Kembalikan pesan segera setelah tersedia.

Untuk informasi tentang cara mengonfirmasi bahwa antrian kosong, lihatMengonfirmasi bahwa antrian Amazon SQS kosong.

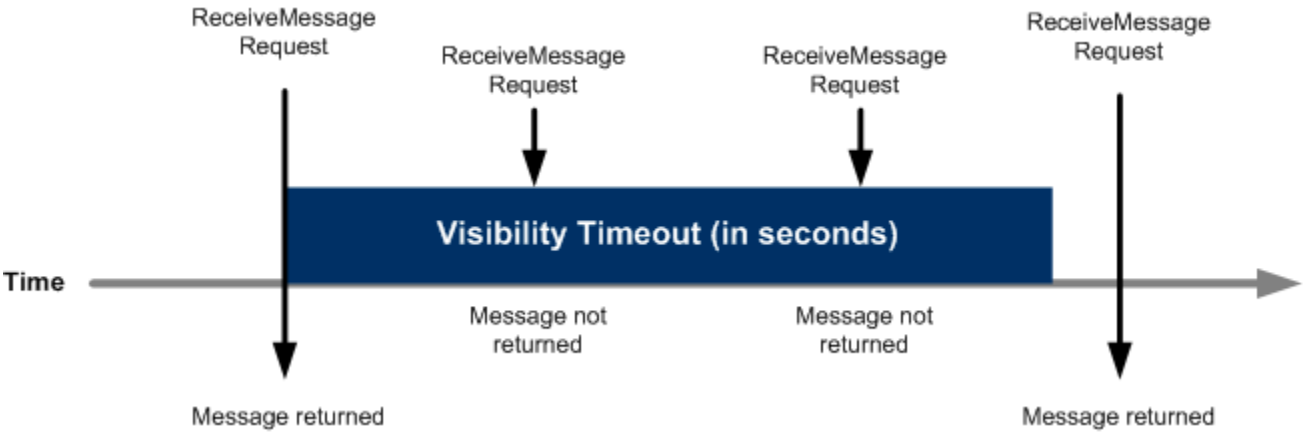
Perbedaan antara polling panjang dan pendek

Polling singkat terjadi ketika `WaitTimeSeconds`parameter `ReceiveMessage` permintaan diatur ke `0` dalam salah satu dari dua cara:

- `ReceiveMessage`Panggilan ditetapkan `WaitTimeSeconds` ke`0`.
- `ReceiveMessage`Panggilan tidak disetel`WaitTimeSeconds`, tetapi atribut antrian `ReceiveMessageWaitTimeSeconds`disetel ke`0`.

Batas waktu visibilitas Amazon SQS

Ketika konsumen menerima dan memproses pesan dari antrian, pesan tetap berada dalam antrian. Amazon SQS tidak menghapus pesan secara otomatis. Karena Amazon SQS adalah sistem terdistribusi, tidak ada jaminan bahwa konsumen benar-benar menerima pesan (misalnya, karena masalah konektivitas, atau karena masalah dalam aplikasi konsumen). Dengan demikian, konsumen harus menghapus pesan dari antrian setelah menerima dan memprosesnya.



Segera setelah pesan diterima, itu tetap dalam antrian. Untuk mencegah konsumen lain memproses pesan lagi, Amazon SQS menetapkan batas waktu visibilitas, periode waktu di mana Amazon SQS mencegah semua konsumen menerima dan memproses pesan. Batas waktu visibilitas default untuk pesan adalah 30 detik. Minimal adalah 0 detik. Maksimal 12 jam. Untuk informasi tentang mengonfigurasi batas waktu visibilitas untuk antrian menggunakan konsol, lihat. Mengkonfigurasi parameter antrian menggunakan konsol Amazon SQS

> **Note**
>
> Untuk antrian standar, batas waktu visibilitas bukanlah jaminan untuk menerima pesan dua kali. Untuk informasi selengkapnya, lihat t-least-once Pengiriman.
> Antrian FIFO memungkinkan produsen atau konsumen untuk mencoba beberapa percobaan ulang:
>
> - Jika produser mendeteksi `SendMessage` tindakan yang gagal, ia dapat mencoba lagi mengirim sebanyak yang diperlukan, menggunakan ID deduplikasi pesan yang sama. Dengan asumsi bahwa produsen menerima setidaknya satu pengakuan sebelum interval deduplikasi berakhir, beberapa percobaan ulang tidak mempengaruhi urutan pesan atau memperkenalkan duplikat.
> - Jika konsumen mendeteksi `ReceiveMessage` tindakan yang gagal, ia dapat mencoba lagi sebanyak yang diperlukan, menggunakan ID percobaan permintaan terima yang sama. Dengan asumsi bahwa konsumen menerima setidaknya satu pengakuan sebelum batas waktu visibilitas berakhir, beberapa percobaan ulang tidak memengaruhi urutan pesan.
> - Saat Anda menerima pesan dengan ID grup pesan, tidak ada lagi pesan untuk ID grup pesan yang sama yang dikembalikan kecuali Anda menghapus pesan atau pesan tersebut menjadi terlihat.

Topik

Dalam pesan penerbangan

Pesan Amazon SQS memiliki tiga status dasar:

1. Dikirim ke antrian oleh produser.
2. Diterima dari antrian oleh konsumen.
3. Dihapus dari antrian.

Sebuah pesan dianggap disimpan setelah dikirim ke antrian oleh produsen, tetapi belum diterima dari antrian oleh konsumen (yaitu, antara negara bagian 1 dan 2). Tidak ada kuota untuk jumlah pesan yang disimpan. Sebuah pesan dianggap dalam penerbangan setelah diterima dari antrian oleh konsumen, tetapi belum dihapus dari antrian (yaitu, antara negara 2 dan 3). Ada kuota untuk jumlah pesan dalam penerbangan.

Important

Kuota yang berlaku untuk pesan dalam penerbangan tidak terkait dengan jumlah pesan tersimpan yang tidak terbatas.

Untuk sebagian besar antrian standar (tergantung pada lalu lintas antrian dan backlog pesan), bisa ada maksimum sekitar 120.000 dalam pesan penerbangan (diterima dari antrian oleh konsumen, tetapi belum dihapus dari antrian). Jika Anda mencapai kuota ini saat menggunakan polling singkat, Amazon SQS mengembalikan `OverLimit` pesan kesalahan. Jika Anda menggunakan polling panjang, Amazon SQS tidak mengembalikan pesan kesalahan. Untuk menghindari mencapai kuota, Anda harus menghapus pesan dari antrian setelah diproses. Anda juga dapat meningkatkan jumlah antrian yang Anda gunakan untuk memproses pesan Anda. Untuk meminta peningkatan kuota, kirimkan permintaan dukungan.

Untuk antrian FIFO, bisa ada maksimum 20.000 dalam pesan penerbangan (diterima dari antrian oleh konsumen, tetapi belum dihapus dari antrian). Jika Anda mencapai kuota ini, Amazon SQS tidak mengembalikan pesan kesalahan.

Important

Saat bekerja dengan antrian FIFO, `DeleteMessage` operasi akan gagal jika permintaan diterima di luar jendela batas waktu visibilitas. Jika batas waktu visibilitas adalah 0 detik, pesan harus dihapus dalam milidetik yang sama saat dikirim, atau dianggap ditinggalkan. Hal

ini dapat menyebabkan Amazon SQS menyertakan pesan duplikat dalam respons yang sama terhadap `ReceiveMessage` operasi jika `MaxNumberOfMessages` parameternya lebih besar dari 1. Untuk detail selengkapnya lihat Cara Kerja Amazon SQS FIFO API.

Mengatur batas waktu visibilitas

Batas waktu visibilitas dimulai saat Amazon SQS mengembalikan pesan. Selama waktu ini, konsumen memproses dan menghapus pesan. Namun, jika konsumen gagal sebelum menghapus pesan dan sistem Anda tidak memanggil `DeleteMessage` tindakan untuk pesan tersebut sebelum batas waktu visibilitas berakhir, pesan akan terlihat oleh konsumen lain dan pesan diterima lagi. Jika pesan harus diterima hanya sekali, konsumen Anda harus menghapusnya dalam durasi batas waktu visibilitas.

Setiap antrian Amazon SQS memiliki pengaturan batas waktu visibilitas default 30 detik. Anda dapat mengubah pengaturan ini untuk seluruh antrian. Biasanya, Anda harus mengatur batas waktu visibilitas ke waktu maksimum yang dibutuhkan aplikasi Anda untuk memproses dan menghapus pesan dari antrian. Saat menerima pesan, Anda juga dapat mengatur batas waktu visibilitas khusus untuk pesan yang dikembalikan tanpa mengubah batas waktu antrian secara keseluruhan. Untuk informasi selengkapnya, lihat praktik terbaik di Memproses pesan tepat waktu bagian ini.

Jika Anda tidak tahu berapa lama waktu yang dibutuhkan untuk memproses pesan, buat detak jantung untuk proses konsumen Anda: Tentukan batas waktu visibilitas awal (misalnya, 2 menit) dan lalu—selama konsumen Anda masih mengerjakan pesan tersebut—terus perpanjang batas waktu visibilitas sebanyak 2 menit setiap menit.

Important

Batas waktu visibilitas maksimum adalah 12 jam dari waktu Amazon SQS menerima permintaan. `ReceiveMessage` Memperpanjang batas waktu visibilitas tidak mengatur ulang maksimum 12 jam.
Selain itu, Anda mungkin tidak dapat mengatur batas waktu pada pesan individual ke 12 jam penuh (misalnya 43.200 detik) sejak `ReceiveMessage` permintaan memulai pengatur waktu. Misalnya, jika Anda menerima pesan dan segera mengatur maksimum 12 jam dengan mengirim `ChangeMessageVisibility` panggilan `VisibilityTimeout` sama dengan 43.200 detik, kemungkinan akan gagal. Namun, menggunakan nilai 43.195 detik akan berfungsi kecuali ada penundaan yang signifikan antara meminta pesan

melalui `ReceiveMessage` dan memperbarui batas waktu visibilitas. Jika konsumen Anda membutuhkan waktu lebih dari 12 jam, pertimbangkan untuk menggunakan Step Functions.

Mengubah batas waktu visibilitas untuk pesan

Saat Anda menerima pesan dari antrian dan mulai memprosesnya, batas waktu visibilitas untuk antrian mungkin tidak mencukupi (misalnya, Anda mungkin perlu memproses dan menghapus pesan). Anda dapat mempersingkat atau memperluas visibilitas pesan dengan menentukan nilai batas waktu baru menggunakan tindakan. `ChangeMessageVisibility`

Misalnya, jika batas waktu default untuk antrian adalah 60 detik, 15 detik telah berlalu sejak Anda menerima pesan, dan Anda mengirim `ChangeMessageVisibility` panggilan dengan `VisibilityTimeout` disetel ke 10 detik, 10 detik mulai dihitung dari waktu Anda melakukan panggilan. `ChangeMessageVisibility` Dengan demikian, setiap upaya untuk mengubah batas waktu visibilitas atau menghapus pesan itu 10 detik setelah Anda awalnya mengubah batas waktu visibilitas (total 25 detik) dapat mengakibatkan kesalahan.

> **Note**
>
> Periode batas waktu baru berlaku sejak Anda memanggil `ChangeMessageVisibility` tindakan. Selain itu, periode batas waktu baru hanya berlaku untuk penerimaan pesan tertentu. `ChangeMessageVisibility`tidak mempengaruhi batas waktu penerimaan pesan selanjutnya atau antrian selanjutnya.

Mengakhiri batas waktu visibilitas untuk pesan

Ketika Anda menerima pesan dari antrian, Anda mungkin menemukan bahwa Anda sebenarnya tidak ingin memproses dan menghapus pesan itu. Amazon SQS memungkinkan Anda menghentikan batas waktu visibilitas untuk pesan tertentu. Ini membuat pesan segera terlihat oleh komponen lain dalam sistem dan tersedia untuk diproses.

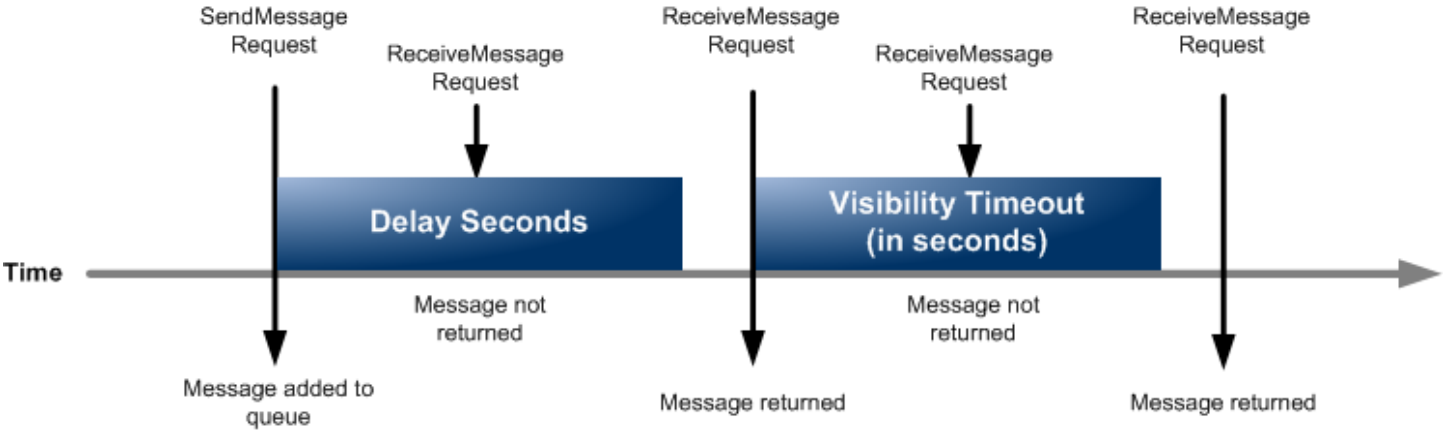
Untuk mengakhiri batas waktu visibilitas pesan setelah meneleponReceiveMessage, panggil `ChangeMessageVisibility`dengan `VisibilityTimeout` disetel ke 0 detik.

Antrian penundaan Amazon SQS

Menunda antrian memungkinkan Anda menunda pengiriman pesan baru ke konsumen selama beberapa detik, misalnya, ketika aplikasi konsumen Anda membutuhkan waktu tambahan untuk memproses pesan. Jika Anda membuat antrean penundaan, pesan apa pun yang Anda kirim ke antrian tetap tidak terlihat oleh konsumen selama periode penundaan. Penundaan default (minimum) untuk antrian adalah 0 detik. Maksimal 15 menit. Untuk informasi tentang mengonfigurasi antrian penundaan menggunakan konsol, lihat. Mengkonfigurasi parameter antrian menggunakan konsol Amazon SQS

> **Note**
>
> Untuk antrian standar, pengaturan penundaan per antrean tidak berlaku surut — mengubah setelan tidak memengaruhi penundaan pesan yang sudah ada dalam antrian.
> Untuk antrian FIFO, pengaturan penundaan per antrean berlaku surut — mengubah pengaturan memengaruhi penundaan pesan yang sudah ada dalam antrian.

Antrian penundaan mirip dengan batas waktu visibilitas karena kedua fitur membuat pesan tidak tersedia bagi konsumen untuk jangka waktu tertentu. Perbedaan antara keduanya adalah bahwa, untuk antrian penundaan, pesan disembunyikan ketika pertama kali ditambahkan ke antrian, sedangkan untuk batas waktu visibilitas pesan disembunyikan hanya setelah dikonsumsi dari antrian. Diagram berikut menggambarkan hubungan antara antrian penundaan dan batas waktu visibilitas.



Untuk mengatur detik penundaan pada pesan individual, bukan pada seluruh antrian, gunakan pengatur waktu pesan untuk mengizinkan Amazon SQS menggunakan nilai pengatur waktu pesan, `DelaySeconds` bukan nilai antrian penundaan. `DelaySeconds`

Antrian sementara Amazon SQS

Antrian sementara membantu Anda menghemat waktu pengembangan dan biaya penerapan saat menggunakan pola pesan umum seperti respon permintaan. Anda dapat menggunakan Klien Antrian Sementara untuk membuat antrean sementara yang dikelola aplikasi dengan throughput tinggi, hemat biaya, dan dikelola.

Klien memetakan beberapa antrian sementara —antrian yang dikelola aplikasi yang dibuat sesuai permintaan untuk proses tertentu—ke satu antrian Amazon SQS secara otomatis. Hal ini memungkinkan aplikasi Anda untuk membuat lebih sedikit panggilan API dan memiliki throughput yang lebih tinggi ketika lalu lintas ke setiap antrian sementara rendah. Ketika antrian sementara tidak lagi digunakan, klien membersihkan antrian sementara secara otomatis, bahkan jika beberapa proses yang menggunakan klien tidak ditutup dengan bersih.

Berikut ini adalah manfaat antrian sementara:

- Mereka berfungsi sebagai saluran komunikasi ringan untuk utas atau proses tertentu.
- Mereka dapat dibuat dan dihapus tanpa menimbulkan biaya tambahan.
- Mereka kompatibel dengan API dengan antrian Amazon SQS statis (normal). Ini berarti bahwa kode yang ada yang mengirim dan menerima pesan dapat mengirim pesan ke dan menerima pesan dari antrian virtual.

Topik

Antrian virtual

Antrian virtual adalah struktur data lokal yang dibuat Klien Antrian Sementara. Antrian virtual memungkinkan Anda menggabungkan beberapa tujuan dengan lalu lintas rendah menjadi satu

antrian Amazon SQS. Untuk praktik terbaik, lihat Hindari menggunakan kembali ID grup pesan yang sama dengan antrian virtual.

> **Note**
>
> - Membuat antrian virtual hanya menciptakan struktur data sementara bagi konsumen untuk menerima pesan. Karena antrian virtual tidak membuat panggilan API ke Amazon SQS, antrian virtual tidak dikenakan biaya.
> - Kuota TPS berlaku untuk semua antrian virtual di satu antrian host. Untuk informasi selengkapnya, lihat Kuota pesan Amazon SQS.

Kelas `AmazonSQSVirtualQueuesClient` pembungkus menambahkan dukungan untuk atribut yang terkait dengan antrian virtual. Untuk membuat antrian virtual, Anda harus memanggil tindakan `CreateQueue` API menggunakan `HostQueueURL` atribut. Atribut ini menentukan antrian yang ada yang menjadi tuan rumah antrian virtual.

URL antrian virtual dalam format berikut.

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

Saat produser memanggil tindakan `SendMessage` atau `SendMessageBatch` API pada URL antrian virtual, Klien Antrian Sementara melakukan hal berikut:

1. Mengekstrak nama antrian virtual.
2. Melampirkan nama antrian virtual sebagai atribut pesan tambahan.
3. Mengirim pesan ke antrian host.

Sementara produser mengirim pesan, thread latar belakang polling antrian host dan mengirim pesan yang diterima ke antrian virtual sesuai dengan atribut pesan yang sesuai.

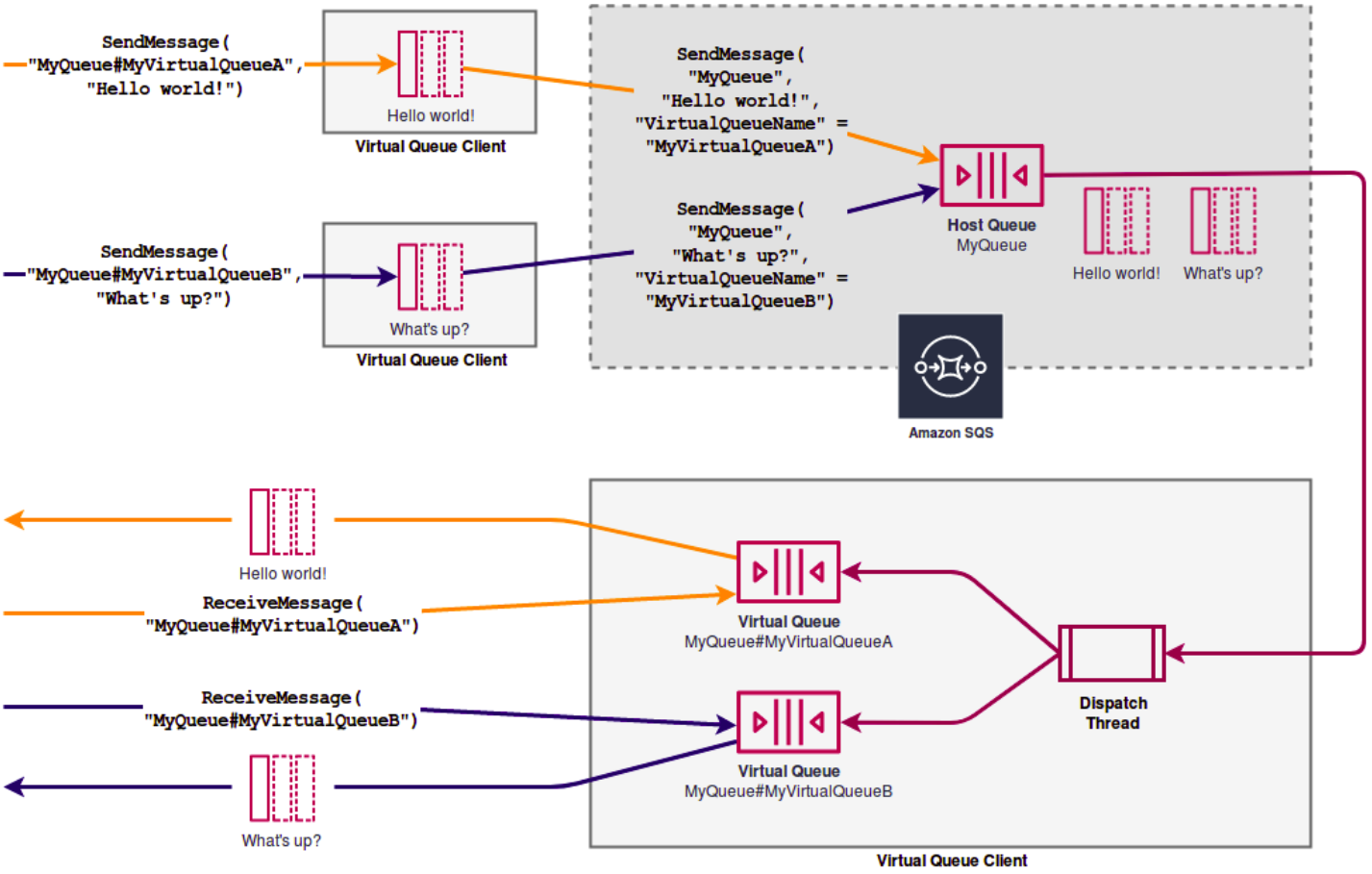
Saat konsumen memanggil tindakan `ReceiveMessage` API pada URL antrian virtual, Klien Antrian Sementara memblokir panggilan secara lokal hingga utas latar belakang mengirimkan pesan ke antrian virtual. (Proses ini mirip dengan pengambilan pesan di Klien Asinkron Buffered: satu tindakan API dapat memberikan pesan hingga 10 antrian virtual.) Menghapus antrian virtual akan menghapus sumber daya sisi klien apa pun tanpa memanggil Amazon SQS itu sendiri.

`AmazonSQSTemporaryQueuesClient`Kelas mengubah semua antrian yang dibuatnya menjadi antrian sementara secara otomatis. Ini juga menciptakan antrian host dengan atribut antrian yang sama secara otomatis, sesuai permintaan. Nama-nama antrian ini berbagi awalan umum yang dapat dikonfigurasi (secara default,`__RequesterClientQueues__`) yang mengidentifikasi mereka sebagai antrian sementara. Hal ini memungkinkan klien untuk bertindak sebagai pengganti drop-in yang mengoptimalkan kode yang ada yang membuat dan menghapus antrian. Klien juga menyertakan `AmazonSQSRequester` dan `AmazonSQSResponder` antarmuka yang memungkinkan komunikasi dua arah antar antrian.

Pola pesan permintaan-respons (antrian virtual)

Kasus penggunaan yang paling umum untuk antrian sementara adalah pola pesan permintaan-respons, di mana pemohon membuat antrian sementara untuk menerima setiap pesan respons. Untuk menghindari pembuatan antrian Amazon SQS untuk setiap pesan respons, Klien Antrian Sementara memungkinkan Anda membuat dan menghapus beberapa antrian sementara tanpa melakukan panggilan API Amazon SQS apa pun. Untuk informasi selengkapnya, lihat Menerapkan sistem permintaan-respons.

Diagram berikut menunjukkan konfigurasi umum menggunakan pola ini.



Contoh skenario: Memproses permintaan login

Contoh skenario berikut menunjukkan bagaimana Anda dapat menggunakan `AmazonSQSRequester` dan `AmazonSQSResponder` antarmuka untuk memproses permintaan login pengguna.

Di sisi klien

```
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();
    }
}
```

Mengirim permintaan login melakukan hal berikut:

1. Membuat antrian sementara.
2. Melampirkan URL antrian sementara ke pesan sebagai atribut.
3. Mengirim pesan.
4. Menerima tanggapan dari antrian sementara.
5. Menghapus antrian sementara.
6. Mengembalikan respon.

Di sisi server

Contoh berikut mengasumsikan bahwa, setelah konstruksi, utas dibuat untuk polling antrian dan memanggil `handleLoginRequest()` metode untuk setiap pesan. Selain itu, `doLogin()` merupakan metode yang diasumsikan.

```
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));
    }
}
```

Membersihkan antrian

Untuk memastikan bahwa Amazon SQS merebut kembali sumber daya dalam memori yang digunakan oleh antrian virtual, ketika aplikasi Anda tidak lagi memerlukan Klien Antrian Sementara, itu harus memanggil metode. `shutdown()` Anda juga dapat menggunakan `shutdown()` metode `AmazonSQSRequester` antarmuka.

Klien Antrian Sementara juga menyediakan cara untuk menghilangkan antrian tuan rumah yatim piatu. Untuk setiap antrian yang menerima panggilan API selama periode waktu tertentu (secara default, lima menit), klien menggunakan tindakan `TagQueue` API untuk menandai antrian yang masih digunakan.

> **Note**
>
> Setiap tindakan API yang diambil pada antrean menandainya sebagai non-idle, termasuk `ReceiveMessage` tindakan yang tidak menampilkan pesan.

Thread latar belakang menggunakan tindakan `ListTags` API `ListQueues` dan untuk memeriksa semua antrian dengan awalan yang dikonfigurasi, menghapus antrian apa pun yang belum diberi tag setidaknya selama lima menit. Dengan cara ini, jika satu klien tidak menutup dengan bersih, klien

aktif lainnya membersihkannya. Untuk mengurangi duplikasi pekerjaan, semua klien dengan awalan yang sama berkomunikasi melalui antrian kerja internal bersama yang dinamai awalan.

Pengatur waktu pesan Amazon SQS

Pengatur waktu pesan memungkinkan Anda menentukan periode tembus pandang awal untuk pesan yang ditambahkan ke antrian. Misalnya, jika Anda mengirim pesan dengan pengatur waktu 45 detik, pesan tersebut tidak terlihat oleh konsumen selama 45 detik pertama dalam antrean. Penundaan default (minimum) untuk pesan adalah 0 detik. Maksimal 15 menit. Untuk informasi tentang mengirim pesan dengan timer menggunakan konsol, lihatKirim pesan.

> **Note**
>
> Antrian FIFO tidak mendukung pengatur waktu pada pesan individual.

Untuk mengatur periode penundaan pada seluruh antrian, bukan pada pesan individual, gunakan antrian penundaan. Pengaturan pengatur waktu pesan untuk pesan individual akan mengganti `DelaySeconds` nilai apa pun pada antrean penundaan Amazon SQS.

Mengakses EventBridge Pipa Amazon melalui konsol Amazon SQS

Amazon EventBridge Pipes menghubungkan sumber ke target. Pipa dimaksudkan untuk point-to-point integrasi antara sumber dan target yang didukung, dengan dukungan untuk transformasi dan pengayaan lanjutan. EventBridge Pipes menyediakan cara yang sangat skalabel untuk menghubungkan antrian Amazon SQS Anda AWS ke layanan seperti Step Functions, Amazon SQS, dan API Gateway, serta aplikasi perangkat lunak pihak ketiga sebagai layanan (SaaS) seperti Salesforce.

Untuk menyiapkan pipa, Anda memilih sumber, menambahkan pemfilteran opsional, menentukan pengayaan opsional, dan memilih target untuk data peristiwa.

Pada halaman detail untuk antrian Amazon SQS, Anda dapat melihat pipa yang menggunakan antrian tersebut sebagai sumbernya. Dari sana, Anda juga bisa:

- Luncurkan EventBridge konsol untuk melihat detail pipa.
- Luncurkan EventBridge konsol untuk membuat pipa baru dengan antrian sebagai sumbernya.

Untuk informasi selengkapnya tentang mengonfigurasi antrian Amazon SQS sebagai sumber pipa, lihat antrian Amazon SQS sebagai sumber di Panduan Pengguna Amazon. EventBridge Untuk informasi lebih lanjut tentang EventBridge Pipa secara umum, lihat EventBridge Pipa.

Untuk mengakses EventBridge pipa untuk antrian Amazon SQS tertentu

1. Buka halaman Antrian konsol Amazon SQS.
2. Pilih antrian.
3. Pada halaman detail antrian, pilih tab EventBridge Pipes.

   Tab EventBridge Pipes menyertakan daftar pipa yang saat ini dikonfigurasi untuk menggunakan antrian yang dipilih sebagai sumber, termasuk:

   - nama pipa
   - status saat ini
   - target pipa
   - ketika pipa terakhir dimodifikasi
4. Lihat lebih banyak detail pipa atau buat pipa baru, jika diinginkan:

   - Untuk mengakses detail lebih lanjut tentang pipa:

     Pilih nama pipa.

     Ini meluncurkan halaman detail Pipe EventBridge konsol.
   - Untuk membuat pipa baru:

     Pilih Connect Amazon SQS antrian ke pipa.

     Ini meluncurkan halaman Create pipe EventBridge konsol, dengan antrean Amazon SQS ditentukan sebagai sumber pipa. Untuk informasi selengkapnya, lihat Membuat EventBridge pipa di Panduan EventBridge Pengguna Amazon.

     > **Important**
     >
     > Pesan pada antrean Amazon SQS dibaca oleh satu pipa dan kemudian dihapus dari antrian setelah diproses, apakah pesan tersebut cocok dengan filter yang dapat Anda konfigurasikan untuk pipa itu atau tidak. Lanjutkan dengan hati-hati

saat mengonfigurasi beberapa pipa untuk menggunakan antrian yang sama dengan sumbernya.

Mengelola pesan Amazon SQS besar dengan Extended Client Library dan Amazon Simple Storage Service

Anda dapat menggunakan Amazon SQS Extended Client Library untuk Java dan Amazon SQS Extended Client Library untuk Python untuk mengirim pesan besar. Ini sangat berguna untuk mengkonsumsi muatan pesan besar, dari 256 KB dan hingga 2 GB. Kedua pustaka menyimpan payload pesan ke bucket Amazon Simple Storage Service, dan mengirim referensi objek Amazon S3 yang disimpan ke antrean Amazon SQS.

> **Note**
>
> Amazon SQS Extended Client Libraries kompatibel dengan antrian Standar dan FIFO.

Topik

- Mengelola pesan Amazon SQS besar menggunakan Java dan Amazon S3
- Mengelola pesan Amazon SQS besar menggunakan Python dan Amazon S3

Mengelola pesan Amazon SQS besar menggunakan Java dan Amazon S3

Anda dapat menggunakan Amazon SQS Extended Client Library untuk Java dan Amazon Simple Storage Service (Amazon S3) Simple Storage Service (Amazon S3) untuk mengelola pesan Amazon Simple Queue Service (Amazon SQS) yang besar. Ini sangat berguna untuk mengkonsumsi muatan pesan besar, dari 256 KB dan hingga 2 GB. Pustaka menyimpan payload pesan ke bucket Amazon S3 dan mengirimkan pesan yang berisi referensi objek Amazon S3 yang disimpan ke antrean Amazon SQS.

Anda dapat menggunakan Amazon SQS Extended Client Library for Java untuk melakukan hal berikut:

- Tentukan apakah pesan selalu disimpan di Amazon S3 atau hanya jika ukuran pesan melebihi 256 KB

- Mengirim pesan yang mereferensikan objek pesan tunggal yang disimpan dalam bucket S3
- Ambil objek pesan dari bucket Amazon S3
- Hapus objek pesan dari bucket Amazon S3

Prasyarat

Contoh berikut menggunakan AWS Java SDK. Untuk menginstal dan menyiapkan SDK, lihat Mengatur AWS SDK for Java di Panduan AWS SDK for Java Pengembang.

Sebelum Anda menjalankan kode contoh, konfigurasikan AWS kredensil Anda. Untuk informasi selengkapnya, lihat Menyiapkan AWS Kredensial dan Wilayah untuk Pengembangan di Panduan AWS SDK for Java Pengembang.

SDK for Java dan Amazon SQS Extended Client Library untuk Java memerlukan J2SE Development Kit 8.0 atau yang lebih baru.

> **Note**
>
> Anda dapat menggunakan Amazon SQS Extended Client Library for Java untuk mengelola pesan Amazon SQS menggunakan Amazon S3 hanya dengan file. AWS SDK for Java Anda tidak dapat melakukan ini dengan AWS CLI, konsol Amazon SQS, Amazon SQS HTTP API, atau SDK lainnya. AWS

AWS SDK for Java 2.x Contoh: Menggunakan Amazon S3 untuk mengelola pesan Amazon SQS besar

Contoh AWS SDK for Java 2.x berikut membuat bucket Amazon S3 dengan nama acak dan menambahkan aturan siklus hidup untuk menghapus objek secara permanen setelah 14 hari. Ini juga membuat antrian bernama `MyQueue` dan mengirim pesan acak yang disimpan dalam ember S3 dan lebih dari 256 KB ke antrian. Akhirnya, kode mengambil pesan, mengembalikan informasi tentangnya, dan kemudian menghapus pesan, antrian, dan ember.

```
/*
 * Copyright 2010-2024 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);
}
}
```

AWS SDK for Java 2.x Contoh: Menggunakan Amazon S3 untuk mengelola pesan Amazon SQS besar

Contoh AWS SDK for Java 2.x berikut membuat bucket Amazon S3 dengan nama acak dan menambahkan aturan siklus hidup untuk menghapus objek secara permanen setelah 14 hari. Ini juga membuat antrian bernama `MyQueue` dan mengirim pesan acak yang disimpan dalam ember S3 dan lebih dari 256 KB ke antrian. Akhirnya, kode mengambil pesan, mengembalikan informasi tentangnya, dan kemudian menghapus pesan, antrian, dan ember.

```
/*
 * Copyright 2010-2024 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());
    }
}
```

Anda dapat menggunakan Apache Maven untuk mengonfigurasi dan membangun Amazon SQS Extended Client untuk proyek Java Anda, atau untuk membangun SDK itu sendiri. Tentukan modul individual dari SDK yang Anda gunakan dalam aplikasi Anda.

```
<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>
```

Mengelola pesan Amazon SQS besar menggunakan Python dan Amazon S3

Anda dapat menggunakan Amazon Simple Queue Service Extended Client Library untuk Python dan Amazon Simple Storage Service untuk mengelola pesan Amazon SQS yang besar. Ini sangat berguna untuk mengkonsumsi muatan pesan besar, dari 256 KB dan hingga 2 GB. Pustaka menyimpan payload pesan ke bucket Amazon S3 dan mengirimkan pesan yang berisi referensi objek Amazon S3 yang disimpan ke antrean Amazon SQS Amazon.

Anda dapat menggunakan Extended Client Library untuk Python untuk melakukan hal berikut:

- Tentukan apakah muatan selalu disimpan di Amazon S3, atau hanya disimpan di S3 jika ukuran muatan melebihi 256 KB
- Kirim pesan yang mereferensikan satu objek pesan yang disimpan di bucket Amazon S3
- Ambil objek payload yang sesuai dari bucket Amazon S3
- Hapus objek payload yang sesuai dari bucket Amazon S3

Prasyarat

Berikut ini adalah prasyarat untuk menggunakan Amazon SQS Extended Client Library untuk Python:

- AWS Akun dengan kredensi yang diperlukan. Untuk membuat AWS akun, navigasikan ke AWS halaman beranda, lalu pilih Buat AWS Akun. Ikuti petunjuk online. Untuk informasi tentang kredensional, lihat Kredensional.

- AWS SDK: Contoh pada halaman ini menggunakan AWS Python SDK Boto3. Untuk menginstal dan menyiapkan SDK, lihat dokumentasi SDK untuk Python di AWS SDK for Python Developer Guide AWS
- Python 3.x (atau yang lebih baru) dan. `pip`
- Perpustakaan Klien Diperpanjang Amazon SQS untuk Python, tersedia dari PyPI

> **Note**
>
> Anda dapat menggunakan Amazon SQS Extended Client Library untuk Python untuk mengelola pesan Amazon SQS menggunakan Amazon S3 hanya dengan SDK untuk Python. AWS Anda tidak dapat melakukan ini dengan AWS CLI, konsol Amazon SQS, Amazon SQS HTTP API, atau SDK lainnya. AWS

Mengkonfigurasi penyimpanan pesan

Amazon SQS Extended Client menggunakan atribut pesan berikut untuk mengonfigurasi opsi penyimpanan pesan Amazon S3:

- `large_payload_support`: Nama bucket Amazon S3 untuk menyimpan pesan besar.
- `always_through_s3`: Jika`True`, maka semua pesan disimpan di Amazon S3. Jika`False`, pesan yang lebih kecil dari 256 KB tidak akan diserialisasikan ke bucket s3. Nilai default-nya `False`.
- `use_legacy_attribute`: Jika`True`, semua pesan yang dipublikasikan menggunakan atribut pesan cadangan Legacy (`SQSLargePayloadSize`), bukan atribut pesan cadangan saat ini (`ExtendedPayloadSize`).

Mengelola pesan Amazon SQS besar dengan Extended Client Library untuk Python

Contoh berikut membuat bucket Amazon S3 dengan nama acak. Kemudian membuat antrian Amazon SQS bernama `MyQueue` dan mengirim pesan yang disimpan dalam bucket S3 dan lebih dari 256 KB ke antrian. Akhirnya, kode mengambil pesan, mengembalikan informasi tentangnya, dan kemudian menghapus pesan, antrian, dan ember.

```
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
```

Mengonfigurasi antrian Amazon SQS menggunakan konsol Amazon SQS

Gunakan konsol Amazon SQS untuk mengonfigurasi dan mengelola antrian dan fitur Amazon Simple Queue Service (Amazon SQS). Anda juga dapat menggunakan konsol untuk mengonfigurasi fitur seperti enkripsi sisi server, mengaitkan antrian huruf mati dengan antrian Anda, atau mengatur pemicu untuk menjalankan fungsi. AWS Lambda

Topik

Kontrol akses berbasis atribut untuk Amazon SQS

Apa itu ABAC?

Attribute-based access control (ABAC) adalah proses otorisasi yang mendefinisikan izin berdasarkan tag yang dilampirkan ke pengguna dan sumber daya. AWS ABAC menyediakan kontrol akses yang terperinci dan fleksibel berdasarkan atribut dan nilai, mengurangi risiko keamanan yang terkait dengan kebijakan berbasis peran yang dikonfigurasi ulang, dan memusatkan audit dan manajemen kebijakan akses. Untuk detail selengkapnya tentang ABAC, lihat Untuk apa ABAC AWS di Panduan Pengguna IAM.

Amazon SQS mendukung ABAC dengan memungkinkan Anda mengontrol akses ke antrian Amazon SQS berdasarkan tag dan alias yang terkait dengan antrian Amazon SQS. Kunci kondisi tag dan

alias yang mengaktifkan ABAC di Amazon SQS memberi otorisasi kepada prinsipal IAM untuk menggunakan antrian Amazon SQS tanpa mengedit kebijakan atau mengelola hibah.

Dengan ABAC, Anda dapat menggunakan tag untuk mengonfigurasi izin dan kebijakan akses IAM untuk antrian Amazon SQS, yang membantu Anda menskalakan pengelolaan izin. Anda dapat membuat kebijakan izin tunggal di IAM menggunakan tag yang ditambahkan ke setiap peran bisnis —tanpa harus memperbarui kebijakan setiap kali menambahkan sumber daya baru. Anda juga dapat melampirkan tag ke prinsipal IAM untuk membuat kebijakan ABAC. Anda dapat mendesain kebijakan ABAC untuk mengizinkan operasi Amazon SQS saat tag pada peran pengguna IAM yang membuat panggilan cocok dengan tag antrean Amazon SQS. Untuk mempelajari selengkapnya tentang penandaan AWS, lihat Strategi AWS Penandaan dan. Tag alokasi biaya Amazon SQS

> **Note**
>
> ABAC untuk Amazon SQS saat ini tersedia di AWS semua Wilayah Komersil di mana Amazon SQS tersedia, dengan pengecualian berikut:
>
> - Asia Pasifik (Hyderabad)
> - Asia Pasifik (Melbourne)
> - Eropa (Spanyol)
> - Eropa (Zürich)

Mengapa saya harus menggunakan ABAC di Amazon SQS?

Berikut adalah beberapa manfaat menggunakan ABAC di Amazon SQS:

- ABAC untuk Amazon SQS memerlukan kebijakan izin yang lebih sedikit. Anda tidak perlu membuat kebijakan yang berbeda untuk fungsi pekerjaan yang berbeda. Anda dapat menggunakan tag sumber daya dan permintaan yang berlaku untuk lebih dari satu antrian, yang mengurangi overhead operasional.
- Gunakan ABAC untuk meningkatkan skala tim dengan cepat. Izin untuk sumber daya baru secara otomatis diberikan berdasarkan tag ketika sumber daya diberi tag dengan tepat selama pembuatannya.
- Gunakan izin pada prinsipal IAM untuk membatasi akses sumber daya. Anda dapat membuat tag untuk prinsipal IAM dan menggunakannya untuk membatasi akses ke tindakan tertentu yang

cocok dengan tag pada prinsipal IAM. Ini membantu Anda mengotomatiskan proses pemberian izin permintaan.

- Lacak siapa yang mengakses sumber daya Anda. Anda dapat menentukan identitas sesi dengan melihat atribut pengguna di AWS CloudTrail.

Topik

Kunci kondisi ABAC untuk Amazon SQS

Anda dapat menggunakan tombol kondisi berikut untuk mengontrol tindakan fungsi:

Kunci syarat ABAC	Deskripsi	Jenis kebijakan	Operasi Amazon SQS
aws: ResourceTag	Tag (kunci dan nilai) pada antrean Amazon SQS cocok dengan tag (kunci dan nilai) atau pola tag dalam kebijakan	Khusus kebijakan IAM	Operasi sumber daya antrian Amazon SQS
aws: RequestTag	Tag (kunci dan nilai) pada operasi sumber daya antrian Amazon SQS cocok dengan tag (kunci dan nilai) atau pola tag dalam kebijakan	Kebijakan antrian dan kebijakan IAM	TagQueue, UntagQueue, CreateQueue
aws: TagKeys	Kunci tag dalam permintaan cocok	Kebijakan antrian dan kebijakan IAM	TagQueue, UntagQueue, CreateQueue

Kunci syarat ABAC	Deskripsi	Jenis kebijakan	Operasi Amazon SQS
	dengan kunci tag dalam kebijakan		

Penandaan untuk kontrol akses di Amazon SQS

Berikut ini adalah contoh cara menggunakan tag untuk kontrol akses. Kebijakan IAM membatasi pengguna IAM untuk semua tindakan Amazon SQS untuk semua antrian yang menyertakan tag sumber daya dengan lingkungan kunci dan produksi nilai. Untuk informasi selengkapnya, lihat Kontrol akses berbasis atribut dengan tag dan Organizations AWS.

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

Membuat pengguna IAM dan antrian Amazon SQS

Contoh berikut menjelaskan cara membuat kebijakan ABAC untuk mengontrol akses ke Amazon SQS menggunakan AWS Management Console dan. AWS CloudFormation

Menggunakan AWS Management Console

Buat pengguna IAM

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.

2. Pilih Pengguna dari panel navigasi kiri.
3. Pilih Tambah Pengguna dan masukkan nama di kotak teks Nama pengguna.
4. Pilih tombol Akses - kotak Akses terprogram dan pilih Berikutnya:Izin.
5. Pilih Selanjutnya: Tag.
6. Tambahkan kunci tag sebagai `environment` dan nilai tag sebagai`beta`.
7. Pilih Berikutnya:Tinjau dan kemudian pilih Buat pengguna.
8. Salin dan simpan ID kunci akses dan kunci akses rahasia di lokasi yang aman.

Tambahkan izin pengguna IAM

1. Pilih pengguna IAM yang Anda buat.
2. Pilih Tambahkan kebijakan inline.
3. Pada tab JSON, tempel kebijakan berikut:

```
{
  "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. Pilih Tinjau kebijakan.
5. Pilih Buat kebijakan.

Menggunakan AWS CloudFormation

Gunakan contoh AWS CloudFormation template berikut untuk membuat pengguna IAM dengan kebijakan inline yang dilampirkan dan antrean 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"
```

Menguji kontrol akses berbasis atribut

Contoh berikut menunjukkan cara menguji kontrol akses berbasis atribut di Amazon SQS.

Buat antrian dengan kunci tag disetel ke lingkungan dan nilai tag disetel ke prod

Jalankan perintah AWS CLI ini untuk menguji pembuatan antrian dengan kunci tag disetel ke lingkungan dan nilai tag disetel ke prod. Jika Anda tidak memiliki AWS CLI, Anda dapat mengunduh dan mengonfigurasinya untuk mesin Anda.

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

Anda menerima `AccessDenied` kesalahan dari titik akhir Amazon SQS:

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

Ini karena nilai tag pada pengguna IAM tidak cocok dengan tag yang diteruskan dalam panggilan `CreateQueue` API. Ingat bahwa kami menerapkan tag ke pengguna IAM dengan kunci yang disetel ke `environment` dan nilai yang disetel ke`beta`.

Buat antrian dengan kunci tag disetel ke lingkungan dan nilai tag disetel ke beta

Jalankan perintah CLI ini untuk menguji pembuatan antrian dengan kunci tag disetel ke `environment` dan nilai tag disetel ke. `beta`

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

Anda menerima pesan yang mengonfirmasi keberhasilan pembuatan antrian, mirip dengan yang di bawah ini.

```
{
"QueueUrl": "<queueUrl>“
}
```

Mengirim pesan ke antrian

Jalankan perintah CLI ini untuk menguji pengiriman pesan ke antrian.

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

Respons menunjukkan pengiriman pesan yang berhasil ke antrean Amazon SQS. Izin pengguna IAM memungkinkan Anda mengirim pesan ke antrian yang memiliki `beta` tag. Tanggapan termasuk `MD5OfMessageBody` dan `MessageId` berisi pesan.

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

Mengkonfigurasi parameter antrian menggunakan konsol Amazon SQS

Saat membuat atau mengedit antrian, Anda dapat mengonfigurasi parameter berikut:

- Batas waktu visibilitas — Lamanya waktu pesan yang diterima dari antrian (oleh satu konsumen) tidak akan terlihat oleh konsumen pesan lainnya. Untuk informasi selengkapnya, lihat Batas waktu visibilitas.

> **Note**
>
> Menggunakan konsol untuk mengonfigurasi batas waktu visibilitas mengonfigurasi nilai batas waktu untuk semua pesan dalam antrian. Untuk mengonfigurasi batas waktu untuk satu atau beberapa pesan, Anda harus menggunakan salah satu AWS SDK.

- Periode penyimpanan pesan — Jumlah waktu Amazon SQS menyimpan pesan yang tetap dalam antrian. Secara default, antrian menyimpan pesan selama empat hari. Anda dapat mengonfigurasi antrian untuk menyimpan pesan hingga 14 hari. Untuk informasi selengkapnya, lihat Periode penyimpanan pesan.
- Penundaan pengiriman - Jumlah waktu yang akan ditunda Amazon SQS sebelum mengirimkan pesan yang ditambahkan ke antrian. Untuk informasi selengkapnya, lihat Penundaan pengiriman.
- Ukuran pesan maksimum - Ukuran pesan maksimum untuk antrian ini. Untuk informasi selengkapnya, lihat Ukuran pesan maksimum.
- Menerima waktu tunggu pesan — Jumlah waktu maksimum Amazon SQS menunggu pesan tersedia setelah antrian mendapat permintaan terima. Untuk informasi selengkapnya, lihat Amazon SQS polling pendek dan panjang.
- Aktifkan deduplikasi berbasis konten — Amazon SQS dapat secara otomatis membuat ID deduplikasi berdasarkan isi pesan. Untuk informasi selengkapnya, lihat Memulai antrian FIFO di Amazon SQS.
- Aktifkan throughput tinggi FIFO — Gunakan untuk mengaktifkan throughput tinggi untuk pesan dalam antrian. Memilih opsi ini mengubah opsi terkait (cakupan Deduplikasi dan batas throughput FIFO) ke pengaturan yang diperlukan untuk mengaktifkan throughput tinggi untuk antrian FIFO. Untuk informasi selengkapnya, lihat Throughput tinggi untuk antrian FIFO di Amazon SQS dan Kuota pesan Amazon SQS.
- Kebijakan redrive allow: mendefinisikan antrian sumber mana yang dapat menggunakan antrian ini sebagai antrian huruf mati. Untuk informasi selengkapnya, lihat Menggunakan antrian huruf mati di Amazon SQS .

Untuk mengonfigurasi parameter antrian untuk antrian yang ada (konsol)

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian. Pilih antrian dan pilih Edit.
3. Gulir ke bagian Konfigurasi.

4. Untuk batas waktu Visibilitas, masukkan durasi dan unit. Kisarannya adalah 0 detik hingga 12 jam. Nilai defaultnya adalah 30 detik.
5. Untuk periode penyimpanan Pesan, masukkan durasi dan unit. Kisarannya adalah 1 menit hingga 14 hari. Nilai default adalah 4 hari.
6. Untuk antrian standar, masukkan nilai untuk Menerima waktu tunggu pesan. Kisarannya adalah 0 hingga 20 detik. Nilai default adalah 0 detik, yang menetapkan polling pendek. Setiap nilai bukan nol menetapkan polling panjang.
7. Untuk keterlambatan Pengiriman, masukkan durasi dan unit. Kisarannya adalah 0 detik hingga 15 menit. Nilai default adalah 0 detik.
8. Untuk Ukuran pesan maksimum, masukkan nilai. Kisarannya adalah 1 KB hingga 256 KB. Nilai defaultnya adalah 256 KB.
9. Untuk antrian FIFO, pilih Aktifkan deduplikasi berbasis konten untuk mengaktifkan deduplikasi berbasis konten. Pengaturan default dinonaktifkan.
10. (Opsional) Untuk antrian FIFO untuk mengaktifkan throughput yang lebih tinggi untuk mengirim dan menerima pesan dalam antrian, pilih Aktifkan FIFO throughput tinggi.

    Memilih opsi ini mengubah opsi terkait (cakupan Deduplikasi dan batas throughput FIFO) ke pengaturan yang diperlukan untuk mengaktifkan throughput tinggi untuk antrian FIFO. Jika Anda mengubah salah satu pengaturan yang diperlukan untuk menggunakan FIFO throughput tinggi, throughput normal berlaku untuk antrian, dan deduplikasi terjadi seperti yang ditentukan. Untuk informasi selengkapnya, lihat Throughput tinggi untuk antrian FIFO di Amazon SQS dan Kuota pesan Amazon SQS.
11. Untuk kebijakan Izinkan Remrive, pilih Diaktifkan. Pilih dari berikut ini: Izinkan semua (default), Dengan antrian atau Tolak semua. Saat memilih Dengan antrian, tentukan daftar hingga 10 antrian sumber berdasarkan Nama Sumber Daya Amazon (ARN).
12. Setelah Anda selesai mengonfigurasi parameter antrian, pilih Simpan.

Mengkonfigurasi kebijakan akses

Saat mengedit antrian, Anda dapat mengonfigurasi kebijakan aksesnya.

Kebijakan akses mendefinisikan akun, pengguna, dan peran yang dapat mengakses antrian. Kebijakan akses juga mendefinisikan tindakan (sepertiSendMessage,ReceiveMessage, atauDeleteMessage) yang dapat diakses pengguna. Kebijakan default hanya mengizinkan pemilik antrian untuk mengirim dan menerima pesan.

Untuk mengonfigurasi kebijakan akses untuk antrean yang ada (konsol)

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian.
3. Pilih antrian dan pilih Edit.
4. Gulir ke bagian Kebijakan akses.
5. Edit pernyataan kebijakan akses di kotak input. Untuk informasi lebih lanjut tentang pernyataan kebijakan akses, lihatManajemen identitas dan akses di Amazon SQS.
6. Setelah selesai mengonfigurasi kebijakan akses, pilih Simpan.

Mengkonfigurasi enkripsi sisi server untuk antrian menggunakan kunci enkripsi yang dikelola SQS

Selain opsi enkripsi sisi server (SSE) terkelola Amazon SQS default, Amazon SQS managed SSE (SSE-SQS) memungkinkan Anda membuat enkripsi sisi server terkelola khusus yang menggunakan kunci enkripsi yang dikelola SQS untuk melindungi data sensitif yang dikirim melalui antrian pesan. Dengan SSE-SQS, Anda tidak perlu membuat dan mengelola kunci enkripsi, atau memodifikasi kode Anda untuk mengenkripsi data Anda. SSE-SQS memungkinkan Anda mengirimkan data dengan aman dan membantu Anda memenuhi kepatuhan enkripsi yang ketat dan persyaratan peraturan tanpa biaya tambahan.

SSE-SQS melindungi data saat istirahat menggunakan enkripsi Advanced Encryption Standard (AES-256) 256-bit. SSE mengenkripsi pesan segera setelah Amazon SQS menerimanya. Amazon SQS menyimpan pesan dalam bentuk terenkripsi dan mendekripsi hanya saat mengirimnya ke konsumen resmi.

> **Note**
>
> - Opsi SSE default hanya efektif ketika Anda membuat antrian tanpa menentukan atribut enkripsi.
> - Amazon SQS memungkinkan Anda mematikan semua enkripsi antrian. Oleh karena itu, mematikan KMS-SSE, tidak akan secara otomatis mengaktifkan SQS-SSE. Jika Anda ingin mengaktifkan SQS-SSE setelah mematikan KMS-SSE, Anda harus menambahkan perubahan atribut dalam permintaan.

Untuk mengkonfigurasi enkripsi SSE-SQS untuk antrian (konsol)

> **Note**
>
> Setiap antrian baru yang dibuat menggunakan titik akhir HTTP (non-TLS) tidak akan mengaktifkan enkripsi SSE-SQS secara default. Ini adalah praktik terbaik keamanan untuk membuat antrian Amazon SQS menggunakan titik akhir HTTPS atau Signature Version 4.

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian.
3. Pilih antrian, lalu pilih Edit.
4. Perluas Enkripsi.
5. Untuk enkripsi sisi Server, pilih Diaktifkan (default).

   > **Note**
   >
   > Dengan SSE diaktifkan, anonim `SendMessage` dan `ReceiveMessage` permintaan ke antrian terenkripsi akan ditolak. Praktik terbaik keamanan Amazon SQS merekomendasikan agar tidak menggunakan permintaan anonim. Jika Anda ingin mengirim permintaan anonim ke antrian Amazon SQS, pastikan untuk menonaktifkan SSE.

6. Pilih kunci Amazon SQS (SSE-SQS). Tidak ada biaya tambahan untuk menggunakan opsi ini.
7. Pilih Simpan.

Mengonfigurasi enkripsi sisi server untuk antrian menggunakan konsol Amazon SQS

Untuk melindungi data dalam pesan antrian, Amazon SQS mengaktifkan enkripsi sisi server (SSE) secara default untuk semua antrian yang baru dibuat. Amazon SQS terintegrasi dengan Amazon Web Services Key Management Service (Amazon Web Services KMS) untuk mengelola kunci KMS untuk enkripsi sisi server (SSE). Untuk informasi tentang menggunakan SSE, lihatEnkripsi saat istirahat di Amazon SQS.

Kunci KMS yang Anda tetapkan ke antrian Anda harus memiliki kebijakan kunci yang menyertakan izin untuk semua kepala sekolah yang berwenang untuk menggunakan antrian. Untuk selengkapnya, lihat Manajemen Kunci.

Jika Anda bukan pemilik kunci KMS, atau jika Anda masuk dengan akun yang tidak memiliki `kms:ListAliases` dan `kms:DescribeKey` izin, Anda tidak akan dapat melihat informasi tentang kunci KMS di konsol Amazon SQS. Mintalah pemilik kunci KMS untuk memberi Anda izin ini. Untuk informasi selengkapnya, lihat Manajemen Kunci.

Saat Anda membuat atau mengedit antrian, Anda dapat mengonfigurasi SSE-KMS.

Untuk mengkonfigurasi SSE-KMS untuk antrian yang ada (konsol)

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian.
3. Pilih antrian, lalu pilih Edit.
4. Perluas Enkripsi.
5. Untuk enkripsi sisi Server, pilih Diaktifkan (default).

> **Note**
>
> Dengan SSE diaktifkan, anonim `SendMessage` dan `ReceiveMessage` permintaan ke antrian terenkripsi akan ditolak. Praktik terbaik keamanan Amazon SQS merekomendasikan agar tidak menggunakan permintaan anonim. Jika Anda ingin mengirim permintaan anonim ke antrian Amazon SQS, pastikan untuk menonaktifkan SSE.

6. Pilih AWS Kunci Layanan Manajemen Kunci (SSE-KMS).

   Konsol menampilkan Deskripsi, Akun, dan ARN kunci KMS dari kunci KMS.
7. Tentukan ID kunci KMS untuk antrian. Untuk informasi selengkapnya, lihat Istilah kunci.

   a. Pilih opsi Pilih alias kunci KMS.

   b. Kunci default adalah kunci KMS terkelola Amazon Web Services untuk Amazon SQS. Untuk menggunakan kunci ini, pilih dari daftar kunci KMS.

   c. Untuk menggunakan kunci KMS kustom dari akun Amazon Web Services Anda, pilih dari daftar kunci KMS. Untuk petunjuk cara membuat kunci KMS kustom, lihat Membuat Kunci di Panduan Pengembang Layanan Manajemen Kunci Amazon Web Services.

   d. Untuk menggunakan kunci KMS kustom yang tidak ada dalam daftar, atau kunci KMS kustom dari akun Amazon Web Services lainnya, pilih Masukkan alias kunci KMS dan masukkan kunci KMS Nama Sumber Daya Amazon (ARN).

8. (Opsional) Untuk periode penggunaan kembali kunci data, tentukan nilai antara 1 menit dan 24 jam. Default adalah 5 menit. Untuk informasi selengkapnya, lihat Memahami periode penggunaan kembali kunci data.
9. Ketika Anda selesai mengkonfigurasi SSE-KMS, pilih Simpan.

Mengonfigurasi tag alokasi biaya untuk antrian menggunakan konsol Amazon SQS

Untuk membantu mengatur dan mengidentifikasi antrian Amazon SQS, Anda dapat menambahkan tag alokasi biaya ke dalamnya. Untuk informasi selengkapnya, lihat Tag alokasi biaya Amazon SQS.

Pada halaman Detail untuk antrian, tab Tagging menampilkan tag untuk antrian.

Saat Anda membuat atau mengedit antrian, Anda dapat mengonfigurasi tag untuk itu.

Untuk mengonfigurasi tag untuk antrian yang ada (konsol)

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian.
3. Pilih antrian dan pilih Edit.
4. Gulir ke bagian Tag.
5. Menambahkan, memodifikasi, atau menghapus tag antrian:

   a. Untuk menambahkan tag, pilih Tambahkan tag baru, masukkan Kunci dan Nilai, lalu pilih Tambahkan tag baru.

   b. Untuk memperbarui tag, ubah Kunci dan Nilainya.

   c. Untuk menghapus tag, pilih Hapus di samping pasangan nilai kunci-nya.

6. Setelah Anda selesai mengonfigurasi tag, pilih Simpan.

Berlangganan antrian ke topik Amazon SNS menggunakan konsol Amazon SQS

Anda dapat berlangganan satu atau beberapa antrian Amazon SQS ke topik Simple Notification Service Amazon (Amazon SNS). Saat Anda mempublikasikan pesan ke suatu topik, Amazon SNS mengirimkan pesan ke setiap antrian berlangganan. Amazon SQS mengelola langganan dan izin apa pun yang diperlukan. Untuk informasi selengkapnya tentang Amazon SNS, lihat Apa itu Amazon SNS? di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon.

Saat Anda berlangganan antrian Amazon SQS ke topik SNS, Amazon SNS menggunakan HTTPS untuk meneruskan pesan ke Amazon SQS. Untuk informasi tentang menggunakan Amazon SNS dengan antrian Amazon SQS terenkripsi, lihat. Konfigurasikan izin KMS untuk layanan AWS

> **Important**
>
> Amazon SQS mendukung maksimal 20 pernyataan per kebijakan akses. Berlangganan topik Amazon SNS menambahkan satu pernyataan seperti itu. Melebihi jumlah ini akan mengakibatkan pengiriman langganan topik gagal.

Untuk berlangganan antrian ke topik SNS (konsol)

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian.
3. Dari daftar antrian, pilih antrian untuk berlangganan topik SNS.
4. Dari Tindakan, pilih Berlangganan topik Amazon SNS.
5. Dari Tentukan topik Amazon SNS yang tersedia untuk menu antrian ini, pilih topik SNS untuk antrian Anda.

   Jika topik SNS tidak tercantum dalam menu, pilih Masukkan topik Amazon SNS ARN, lalu masukkan nama sumber daya Amazon (ARN) topik tersebut.
6. Pilih Simpan.
7. Untuk memverifikasi hasil langganan, publikasikan ke topik dan kemudian lihat pesan yang dikirim topik ke antrian. Untuk informasi selengkapnya, lihat Penerbitan pesan Amazon SNS di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon.

Jika antrian Amazon SQS dan topik SNS Anda berbeda Akun AWS, pemilik topik harus terlebih dahulu mengonfirmasi langganan. Untuk informasi selengkapnya, lihat Konfirmasi langganan di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon.

Untuk informasi tentang berlangganan topik SNS lintas wilayah, lihat Mengirim pesan Amazon SNS ke antrean atau fungsi Amazon SQS di Wilayah lain AWS Lambda di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon

Mengonfigurasi antrian Amazon SQS untuk memicu fungsi AWS Lambda

Anda dapat menggunakan AWS Lambda fungsi untuk memproses pesan dalam antrean Amazon SQS. Lambda melakukan polling pada antrean dan memanggil fungsi Lambda Anda secara sinkron dengan kejadian yang berisi pesan antrean. Untuk memungkinkan waktu fungsi Anda memproses setiap kumpulan rekaman, atur batas waktu visibilitas antrian sumber menjadi setidaknya enam kali batas waktu yang Anda konfigurasikan pada fungsi Anda. Waktu tambahan memungkinkan Lambda untuk mencoba lagi jika fungsi Anda dibatasi saat memproses batch sebelumnya.

Anda dapat menentukan antrian lain untuk bertindak sebagai antrean huruf mati untuk pesan yang tidak dapat diproses oleh fungsi Lambda Anda.

Fungsi Lambda dapat memproses item dari beberapa antrian (menggunakan satu sumber peristiwa Lambda untuk setiap antrian). Anda dapat menggunakan antrian yang sama dengan beberapa fungsi Lambda.

Jika Anda mengaitkan antrian terenkripsi dengan fungsi Lambda tetapi Lambda tidak melakukan polling untuk pesan, tambahkan `kms:Decrypt` izin tersebut ke peran eksekusi Lambda Anda.

Perhatikan batasan berikut:

- Antrian Anda dan fungsi Lambda harus berada di Wilayah yang AWS sama.
- Antrian terenkripsi yang menggunakan kunci default (kunci KMS AWS terkelola untuk Amazon SQS) tidak dapat menjalankan fungsi Lambda secara berbeda. Akun AWS

Untuk informasi tentang penerapan fungsi Lambda, lihat Menggunakan AWS Lambda dengan Amazon SQS di AWS Lambda Panduan Pengembang.

Prasyarat

Untuk mengonfigurasi pemicu fungsi Lambda, Anda harus memenuhi persyaratan berikut:

- Jika Anda menggunakan pengguna, peran Amazon SQS Anda harus menyertakan izin berikut:
  - `lambda:CreateEventSourceMapping`
  - `lambda:ListEventSourceMappings`
  - `lambda:ListFunctions`
- Peran eksekusi Lambda harus menyertakan izin berikut:
  - `sqs:DeleteMessage`
  - `sqs:GetQueueAttributes`
  - `sqs:ReceiveMessage`
- Jika Anda mengaitkan antrian terenkripsi dengan fungsi Lambda, tambahkan izin `kms:Decrypt` ke peran eksekusi Lambda.

Untuk informasi selengkapnya, lihat Ikhtisar mengelola akses di Amazon SQS.

Untuk mengonfigurasi antrian untuk memicu fungsi Lambda (konsol)

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian.
3. Pada halaman Antrian, pilih antrian untuk dikonfigurasi.
4. Pada halaman antrian, pilih tab pemicu Lambda.
5. Pada halaman pemicu Lambda, pilih pemicu Lambda.

   Jika daftar tidak menyertakan pemicu Lambda yang Anda butuhkan, pilih Konfigurasi pemicu fungsi Lambda. Masukkan Nama Sumber Daya Amazon (ARN) dari fungsi Lambda atau pilih sumber daya yang ada. Lalu, pilih Simpan.
6. Pilih Simpan. Konsol menyimpan konfigurasi dan menampilkan halaman Detail untuk antrian.

   Pada halaman Detail, tab pemicu Lambda menampilkan fungsi Lambda dan statusnya. Dibutuhkan sekitar 1 menit agar fungsi Lambda dikaitkan dengan antrian Anda.
7. Untuk memverifikasi hasil konfigurasi, kirim pesan ke antrian Anda, lalu lihat fungsi Lambda yang dipicu di konsol Lambda.

Mengotomatiskan pemberitahuan dari AWS layanan ke Amazon SQS menggunakan Amazon EventBridge

Amazon EventBridge memungkinkan Anda mengotomatiskan AWS layanan dan merespons peristiwa sistem seperti masalah ketersediaan aplikasi atau perubahan sumber daya. Acara dari AWS layanan dikirim ke EventBridge hampir secara real time. Anda dapat menuliskan aturan sederhana untuk menunjukkan peristiwa mana yang sesuai kepentingan Anda, dan tindakan otomatis yang diambil ketika suatu peristiwa sesuai dengan suatu aturan.

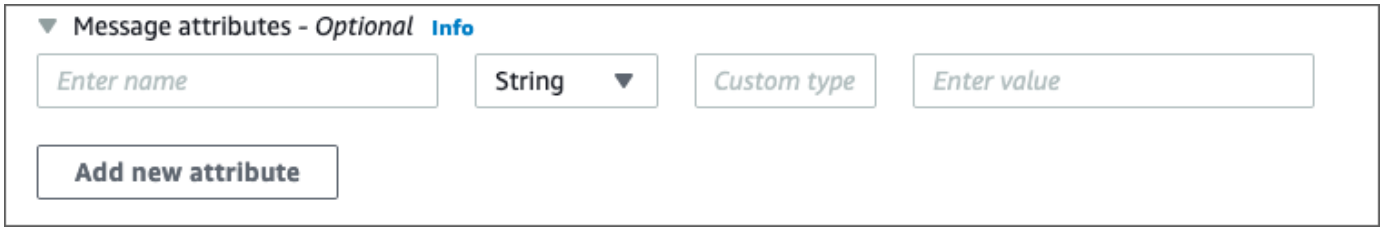
EventBridge memungkinkan Anda menetapkan berbagai target - seperti standar Amazon SQS dan antrian FIFO - yang menerima peristiwa dalam format JSON. Untuk informasi selengkapnya, lihat EventBridgeTarget Amazon di Panduan EventBridge Pengguna Amazon.

Mengirim pesan dengan atribut

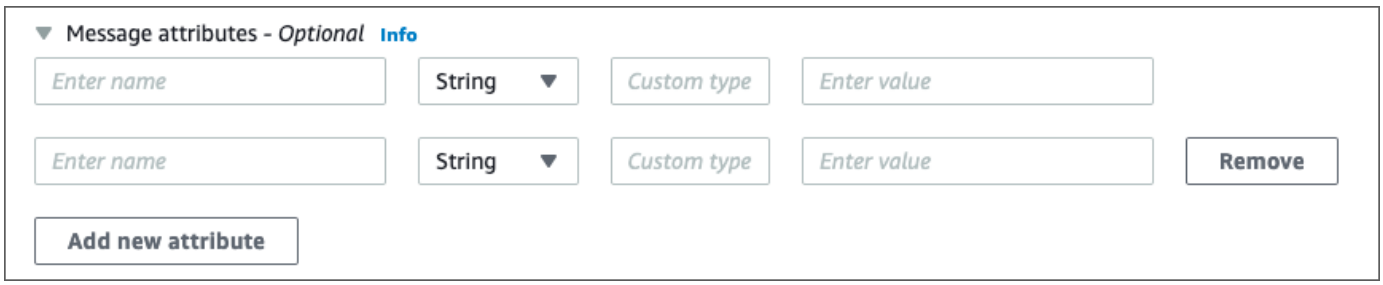
Untuk antrian standar dan FIFO, Anda dapat menyertakan metadata terstruktur (seperti stempel waktu, data geospasial, tanda tangan, dan pengidentifikasi) dengan pesan. Untuk informasi selengkapnya, lihat Atribut pesan Amazon SQS.

Untuk mengirim pesan dengan atribut ke antrian menggunakan konsol Amazon SQS

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Di panel navigasi, pilih Antrian.
3. Pada halaman Antrian, pilih antrian.
4. Pilih Kirim dan terima pesan.
5. Masukkan parameter atribut pesan.
    a. Di kotak teks nama, masukkan nama unik hingga 256 karakter.
    b. Untuk jenis atribut, pilih String, Number, atau Binary.
    c. (Opsional) Masukkan tipe data khusus. Misalnya, Anda dapat menambahkan**`byte`**,**`int`**, atau **`float`** sebagai tipe data khusus untuk Nomor.
    d. Di kotak teks nilai, masukkan nilai atribut pesan.



6. Untuk menambahkan atribut pesan lain., pilih Tambahkan atribut baru.



7. Anda dapat mengubah nilai atribut kapan saja sebelum mengirim pesan.
8. Untuk menghapus atribut, pilih Hapus. Untuk menghapus atribut pertama, tutup atribut Pesan.
9. Setelah selesai menambahkan atribut ke pesan, pilih Kirim pesan. Pesan Anda dikirim dan konsol menampilkan pesan sukses. Untuk melihat informasi tentang atribut pesan dari pesan terkirim, pilih Lihat detail. Pilih Selesai untuk menutup kotak dialog Rincian pesan.

Praktik terbaik untuk Amazon SQS

Praktik terbaik ini dapat membantu Anda memaksimalkan Amazon SQS.

Topik

Rekomendasi untuk standar Amazon SQS dan antrian FIFO

Praktik terbaik berikut dapat membantu Anda mengurangi biaya dan memproses pesan secara efisien menggunakan Amazon SQS.

Topik

Bekerja dengan pesan Amazon SQS

Panduan berikut dapat membantu Anda memproses pesan secara efisien menggunakan Amazon SQS.

Topik

Memproses pesan tepat waktu

Pengaturan batas waktu visibilitas tergantung pada berapa lama waktu yang dibutuhkan aplikasi Anda untuk memproses dan menghapus pesan. Misalnya, jika aplikasi Anda memerlukan 10 detik untuk memproses pesan dan Anda mengatur batas waktu visibilitas menjadi 15 menit, Anda harus menunggu waktu yang relatif lama untuk mencoba memproses pesan lagi jika upaya pemrosesan sebelumnya gagal. Atau, jika aplikasi Anda memerlukan 10 detik untuk memproses pesan tetapi Anda menetapkan batas waktu visibilitas hanya 2 detik, pesan duplikat diterima oleh konsumen lain saat konsumen asli masih mengerjakan pesan.

Untuk memastikan bahwa ada cukup waktu untuk memproses pesan, gunakan salah satu strategi berikut:

- Jika Anda tahu (atau dapat memperkirakan secara wajar) berapa lama waktu yang dibutuhkan untuk memproses pesan, perpanjang batas waktu visibilitas pesan ke waktu maksimum yang diperlukan untuk memproses dan menghapus pesan. Untuk informasi selengkapnya, lihat Mengonfigurasi Batas Waktu Visibilitas.
- Jika Anda tidak tahu berapa lama waktu yang dibutuhkan untuk memproses pesan, buat detak jantung untuk proses konsumen Anda: Tentukan batas waktu visibilitas awal (misalnya, 2 menit) dan lalu—selama konsumen Anda masih mengerjakan pesan tersebut—terus perpanjang batas waktu visibilitas sebanyak 2 menit setiap menit.

> **Important**
>
> Batas waktu visibilitas maksimum adalah 12 jam dari waktu Amazon SQS menerima permintaan. `ReceiveMessage` Memperpanjang batas waktu visibilitas tidak mengatur ulang maksimum 12 jam.
> Selain itu, Anda mungkin tidak dapat mengatur batas waktu pada pesan individual ke 12 jam penuh (misalnya 43.200 detik) sejak `ReceiveMessage` permintaan memulai pengatur waktu. Misalnya, jika Anda menerima pesan dan segera mengatur maksimum 12 jam dengan mengirim `ChangeMessageVisibility` panggilan `VisibilityTimeout` sama dengan 43.200 detik, kemungkinan akan gagal. Namun, menggunakan nilai 43.195 detik akan berfungsi kecuali ada penundaan yang signifikan antara meminta pesan melalui `ReceiveMessage` dan memperbarui batas waktu visibilitas. Jika konsumen Anda membutuhkan waktu lebih dari 12 jam, pertimbangkan untuk menggunakan Step Functions.

Menangani kesalahan permintaan

Untuk menangani kesalahan permintaan, gunakan salah satu strategi berikut:

- Jika Anda menggunakan AWS SDK, Anda sudah memiliki logika coba ulang dan backoff otomatis yang Anda inginkan. Untuk informasi selengkapnya, lihat Error Retries dan Exponential Backoff di. AWSReferensi Umum Amazon Web Services
- Jika Anda tidak menggunakan fitur AWS SDK untuk coba lagi dan backoff, izinkan jeda (misalnya, 200 ms) sebelum mencoba kembali ReceiveMessagetindakan setelah tidak menerima pesan, batas waktu, atau pesan kesalahan dari Amazon SQS. Untuk penggunaan selanjutnya `ReceiveMessage` yang memberikan hasil yang sama, biarkan jeda yang lebih lama (misalnya, 400 ms).

Menyiapkan polling panjang

Ketika waktu tunggu untuk tindakan `ReceiveMessage` API lebih besar dari 0, polling panjang berlaku. Waktu tunggu polling maksimum yang panjang adalah 20 detik. Polling panjang membantu mengurangi biaya penggunaan Amazon SQS dengan menghilangkan jumlah respons kosong (bila tidak ada pesan yang tersedia untuk `ReceiveMessage` permintaan) dan respons kosong palsu (saat pesan tersedia tetapi tidak disertakan dalam respons). Untuk informasi selengkapnya, lihat Amazon SQS polling pendek dan panjang.

Untuk pemrosesan pesan yang optimal, gunakan strategi berikut:

- Dalam kebanyakan kasus, Anda dapat mengatur waktu `ReceiveMessage` tunggu menjadi 20 detik. Jika 20 detik terlalu lama untuk aplikasi Anda, tetapkan waktu `ReceiveMessage` tunggu yang lebih pendek (minimum 1 detik). Jika Anda tidak menggunakan AWS SDK untuk mengakses Amazon SQS, atau jika Anda mengonfigurasi SDK agar memiliki waktu tunggu AWS yang lebih singkat, Anda mungkin harus memodifikasi klien Amazon SQS untuk mengizinkan permintaan yang lebih lama atau menggunakan waktu tunggu yang lebih singkat untuk polling yang lama.
- Jika Anda menerapkan polling panjang untuk beberapa antrian, gunakan satu utas untuk setiap antrian, bukan satu utas untuk semua antrian. Menggunakan satu utas untuk setiap antrian memungkinkan aplikasi Anda memproses pesan di setiap antrian saat tersedia, sementara menggunakan satu utas untuk polling beberapa antrian dapat menyebabkan aplikasi Anda tidak dapat memproses pesan yang tersedia di antrian lain saat aplikasi menunggu (hingga 20 detik) untuk antrian yang tidak memiliki pesan yang tersedia.

> **Important**
>
> Untuk menghindari kesalahan HTTP, pastikan batas waktu respons HTTP untuk `ReceiveMessage` permintaan lebih panjang dari `WaitTimeSeconds` parameter. Untuk informasi lebih lanjut, lihat ReceiveMessage.

Menangkap pesan bermasalah

Untuk menangkap semua pesan yang tidak dapat diproses, dan untuk mengumpulkan CloudWatch metrik yang akurat, konfigurasikan antrian huruf mati.

- Kebijakan redrive mengalihkan pesan ke antrean huruf mati setelah antrian sumber gagal memproses pesan beberapa kali tertentu.
- Menggunakan antrian surat mati mengurangi jumlah pesan dan mengurangi kemungkinan mengekspos Anda ke pesan pil racun (pesan yang diterima tetapi tidak dapat diproses).
- Termasuk pesan pil racun dalam antrian dapat mendistorsi `ApproximateAgeOfOldestMessage` CloudWatch metrik dengan memberikan usia yang salah dari pesan pil racun. Mengkonfigurasi antrian huruf mati membantu menghindari alarm palsu saat menggunakan metrik ini.

Mengatur retensi antrian surat mati

Untuk antrian standar, kedaluwarsa pesan selalu didasarkan pada stempel waktu enqueue aslinya. Ketika pesan dipindahkan ke antrian huruf mati, stempel waktu enqueue tidak berubah. `ApproximateAgeOfOldestMessage`Metrik menunjukkan kapan pesan dipindahkan ke antrian huruf mati, bukan saat pesan awalnya dikirim. Misalnya, asumsikan bahwa pesan menghabiskan 1 hari dalam antrian asli sebelum dipindahkan ke antrian huruf mati. Jika periode retensi antrian surat mati adalah 4 hari, pesan akan dihapus dari antrian surat mati setelah 3 hari dan 3 hari. `ApproximateAgeOfOldestMessage` Dengan demikian, ini adalah praktik terbaik untuk selalu mengatur periode retensi antrian huruf mati menjadi lebih lama dari periode retensi antrian asli.

Untuk antrian FIFO, stempel waktu enqueue akan disetel ulang saat pesan dipindahkan ke antrian huruf mati. `ApproximateAgeOfOldestMessage`Metrik menunjukkan kapan pesan dipindahkan ke antrian huruf mati. Dalam contoh yang sama di atas, pesan dihapus dari antrian huruf mati setelah 4 hari dan `ApproximateAgeOfOldestMessage` adalah 4 hari.

Menghindari pemrosesan pesan yang tidak konsisten

Karena Amazon SQS adalah sistem terdistribusi, adalah mungkin bagi konsumen untuk tidak menerima pesan bahkan ketika Amazon SQS menandai pesan sebagai terkirim saat berhasil kembali dari `ReceiveMessage` panggilan metode API. Dalam hal ini, Amazon SQS mencatat pesan seperti yang dikirimkan setidaknya sekali, meskipun konsumen belum pernah menerimanya. Karena tidak ada upaya tambahan untuk mengirimkan pesan dalam kondisi ini, kami tidak menyarankan menyetel jumlah penerimaan maksimum menjadi 1 untuk antrian huruf mati.

Menerapkan sistem permintaan-respons

Saat menerapkan sistem request-response atau remote procedure call (RPC), ingatlah praktik terbaik berikut ini:

- Jangan membuat antrian balasan per pesan. Sebagai gantinya, buat antrian balasan saat start-up, per produser, dan gunakan atribut pesan ID korelasi untuk memetakan balasan ke permintaan.
- Jangan biarkan produsen Anda berbagi antrian balasan. Hal ini dapat menyebabkan produser menerima pesan respons yang ditujukan untuk produser lain.

Untuk informasi selengkapnya tentang penerapan pola request-response menggunakan Temporary Queue Client, lihat. Pola pesan permintaan-respons (antrian virtual)

Mengurangi biaya Amazon SQS

Praktik terbaik berikut dapat membantu Anda mengurangi biaya dan memanfaatkan potensi pengurangan biaya tambahan dan respons yang hampir seketika.

Tindakan pesan batching

Untuk mengurangi biaya, kumpulkan tindakan pesan Anda:

- Untuk mengirim, menerima, dan menghapus pesan, serta mengubah batas waktu visibilitas pesan untuk beberapa pesan dengan satu tindakan, gunakan tindakan API batch Amazon SQS.
- Untuk menggabungkan buffering sisi klien dengan batch permintaan, gunakan polling panjang bersama dengan klien asinkron buffer yang disertakan dengan. AWS SDK for Java

> **Note**
>
> Klien Asinkron Buffered Amazon SQS saat ini tidak mendukung antrian FIFO.

Menggunakan modus polling yang sesuai

- Polling panjang memungkinkan Anda mengkonsumsi pesan dari antrean Amazon SQS segera setelah tersedia.
  - Untuk mengurangi biaya penggunaan Amazon SQS dan mengurangi jumlah penerima kosong ke antrian kosong (tanggapan terhadap `ReceiveMessage` tindakan yang tidak menampilkan pesan), aktifkan polling panjang. Untuk informasi selengkapnya, lihat Polling Panjang Amazon SQS.
  - Untuk meningkatkan efisiensi saat melakukan polling untuk beberapa utas dengan beberapa penerima, kurangi jumlah utas.
  - Jajak pendapat panjang lebih disukai daripada polling pendek dalam banyak kasus.
- Polling singkat segera mengembalikan respons, bahkan jika antrian Amazon SQS yang disurvei kosong.
  - Untuk memenuhi persyaratan aplikasi yang mengharapkan tanggapan segera terhadap `ReceiveMessage` permintaan, gunakan jajak pendapat singkat.
  - Jajak pendapat pendek ditagih sama dengan long polling.

Pindah dari antrian Amazon SQS Standard ke antrian FIFO

Jika Anda tidak menyetel `DelaySeconds` parameter pada setiap pesan, Anda dapat pindah ke antrean FIFO dengan memberikan ID grup pesan untuk setiap pesan terkirim.

Untuk informasi selengkapnya, lihat Pindah dari antrian standar ke antrian FIFO di Amazon SQS.

Rekomendasi tambahan untuk antrian Amazon SQS FIFO

Praktik terbaik berikut dapat membantu Anda menggunakan ID deduplikasi pesan dan ID grup pesan secara optimal. Untuk informasi selengkapnya, lihat `SendMessage`dan `SendMessageBatch`tindakan di Referensi API Layanan Antrian Sederhana Amazon.

Topik

Menggunakan ID deduplikasi pesan Amazon SQS

ID deduplikasi pesan adalah token yang digunakan untuk deduplikasi pesan terkirim. Jika pesan dengan ID deduplikasi pesan tertentu berhasil dikirim, pesan apa pun yang dikirim dengan ID deduplikasi pesan yang sama berhasil diterima tetapi tidak dikirim selama interval deduplikasi 5 menit.

> **Note**
>
> Amazon SQS terus melacak ID deduplikasi pesan bahkan setelah pesan diterima dan dihapus.

Memberikan ID deduplikasi pesan

Produser harus memberikan nilai ID deduplikasi pesan untuk setiap pesan dalam skenario berikut:

- Pesan yang dikirim dengan badan pesan identik yang harus diperlakukan oleh Amazon SQS sebagai unik.
- Pesan yang dikirim dengan konten yang identik tetapi atribut pesan berbeda yang harus diperlakukan oleh Amazon SQS sebagai unik.
- Pesan yang dikirim dengan konten yang berbeda (misalnya, hitungan coba lagi yang disertakan dalam badan pesan) yang harus diperlakukan Amazon SQS sebagai duplikat.

Mengaktifkan deduplikasi untuk sistem produsen/konsumen tunggal

Jika Anda memiliki satu produsen dan satu konsumen dan pesannya unik karena ID pesan khusus aplikasi disertakan dalam isi pesan, ikuti praktik terbaik berikut ini:

- Aktifkan deduplikasi berbasis konten untuk antrian (setiap pesan Anda memiliki badan yang unik). Produser dapat menghilangkan ID deduplikasi pesan.

- Saat deduplikasi berbasis konten diaktifkan untuk antrean Amazon SQS FIFO, dan pesan dikirim dengan ID deduplikasi, ID deduplikasi akan mengganti ID deduplikasi berbasis konten yang dihasilkan. `SendMessage`
- Meskipun konsumen tidak diharuskan untuk memberikan ID percobaan permintaan terima untuk setiap permintaan, ini adalah praktik terbaik karena memungkinkan urutan coba gagal untuk dijalankan lebih cepat.
- Anda dapat mencoba lagi mengirim atau menerima permintaan karena tidak mengganggu urutan pesan dalam antrian FIFO.

Merancang untuk skenario pemulihan pemadaman

Proses deduplikasi dalam antrian FIFO sensitif terhadap waktu. Saat merancang aplikasi Anda, pastikan bahwa produsen dan konsumen dapat pulih jika terjadi pemadaman klien atau jaringan.

- Produsen harus menyadari interval deduplikasi antrian. Amazon SQS memiliki interval deduplikasi 5 menit. Mencoba kembali `SendMessage` permintaan setelah interval deduplikasi berakhir dapat memperkenalkan pesan duplikat ke dalam antrian. Misalnya, perangkat seluler di dalam mobil mengirim pesan yang pesanannya penting. Jika mobil kehilangan konektivitas seluler untuk jangka waktu tertentu sebelum menerima pengakuan, mencoba kembali permintaan setelah mendapatkan kembali konektivitas seluler dapat membuat duplikat.
- Konsumen harus memiliki batas waktu visibilitas yang meminimalkan risiko tidak dapat memproses pesan sebelum batas waktu visibilitas berakhir. Anda dapat memperpanjang batas waktu visibilitas saat pesan sedang diproses dengan memanggil tindakan. `ChangeMessageVisibility` Namun, jika batas waktu visibilitas berakhir, konsumen lain dapat segera mulai memproses pesan, menyebabkan pesan diproses beberapa kali. Untuk menghindari skenario ini, konfigurasikan antrian huruf mati.

Bekerja dengan batas waktu visibilitas

Untuk kinerja optimal, atur batas waktu visibilitas menjadi lebih besar dari batas waktu baca AWS SDK. Ini berlaku untuk menggunakan aksi `ReceiveMessage` API dengan polling pendek atau polling panjang.

Menggunakan ID grup pesan Amazon SQS

`MessageGroupId`adalah tag yang menentukan bahwa pesan milik grup pesan tertentu. Pesan yang termasuk dalam grup pesan yang sama selalu diproses satu per satu, dalam urutan yang ketat

relatif terhadap grup pesan (namun, pesan yang termasuk dalam grup pesan yang berbeda mungkin diproses secara tidak berurutan).

Menginterleaving beberapa grup pesan yang dipesan

Untuk menginterleave beberapa grup pesan yang diurutkan dalam satu antrian FIFO, gunakan nilai ID grup pesan (misalnya, data sesi untuk beberapa pengguna). Dalam skenario ini, beberapa konsumen dapat memproses antrian, tetapi data sesi setiap pengguna diproses dengan cara FIFO.

> **Note**
>
> Jika pesan yang termasuk dalam ID grup pesan tertentu tidak terlihat, tidak ada konsumen lain yang dapat memproses pesan dengan ID grup pesan yang sama.

Menghindari pemrosesan duplikat dalam sistem multi-produsen/konsumen

Untuk menghindari pemrosesan pesan duplikat dalam sistem dengan beberapa produsen dan konsumen di mana throughput dan latensi lebih penting daripada pemesanan, produsen harus menghasilkan ID grup pesan unik untuk setiap pesan.

> **Note**
>
> Dalam skenario ini, duplikat dihilangkan. Namun, pemesanan pesan tidak dapat dijamin. Skenario apa pun dengan beberapa produsen dan konsumen meningkatkan risiko mengirimkan pesan duplikat secara tidak sengaja jika pekerja tidak memproses pesan dalam batas waktu visibilitas dan pesan menjadi tersedia untuk pekerja lain.

Hindari memiliki backlog besar pesan dengan ID grup pesan yang sama

Untuk antrian FIFO, bisa ada maksimum 20.000 dalam pesan penerbangan (diterima dari antrian oleh konsumen, tetapi belum dihapus dari antrian). Jika Anda mencapai kuota ini, Amazon SQS tidak mengembalikan pesan kesalahan. Antrean FIFO memeriksa 20k pesan pertama untuk menentukan grup pesan yang tersedia. Ini berarti bahwa jika Anda memiliki backlog pesan dalam satu grup pesan, Anda tidak dapat menggunakan pesan dari grup pesan lain yang dikirim ke antrian di lain waktu hingga Anda berhasil menggunakan pesan dari backlog.

> **Note**
>
> Backlog pesan yang memiliki ID grup pesan yang sama mungkin terbentuk karena konsumen yang tidak berhasil memproses pesan. Masalah pemrosesan pesan dapat terjadi karena masalah dengan konten pesan atau karena masalah teknis dengan konsumen.
> Untuk memindahkan pesan yang tidak dapat diproses berulang kali, dan untuk membuka blokir pemrosesan pesan lain yang memiliki ID grup pesan yang sama, pertimbangkan untuk menyiapkan kebijakan antrian huruf mati.

Hindari menggunakan kembali ID grup pesan yang sama dengan antrian virtual

Untuk mencegah pesan dengan ID grup pesan yang sama dikirim ke antrian virtual yang berbeda dengan antrian host yang sama saling memblokir, hindari menggunakan kembali ID grup pesan yang sama dengan antrian virtual.

Menggunakan ID percobaan permintaan terima Amazon SQS

ID percobaan permintaan terima adalah token yang digunakan untuk deduplikasi panggilan. `ReceiveMessage`

Selama pemadaman jaringan jangka panjang yang menyebabkan masalah konektivitas antara SDK dan Amazon SQS, sebaiknya berikan ID percobaan permintaan terima dan coba lagi dengan ID percobaan permintaan terima yang sama jika operasi SDK gagal.

Contoh Amazon SQS Java SDK

Anda dapat menggunakan AWS SDK for Java untuk membangun aplikasi Java yang berinteraksi dengan Amazon Simple Queue Service (Amazon SQS) dan layanan lainnya. AWS Untuk menginstal dan menyiapkan SDK, lihat Memulai di Panduan AWS SDK for Java 2.x Pengembang.

Untuk contoh operasi antrean Amazon SQS dasar, seperti cara membuat antrian atau mengirim pesan, lihat Bekerja dengan Antrian Pesan Amazon SQS di Panduan Pengembang.AWS SDK for Java 2.x

Contoh dalam topik ini menunjukkan fitur Amazon SQS tambahan, seperti enkripsi sisi server (SSE), tag alokasi biaya, dan atribut pesan.

Topik

- Menggunakan enkripsi sisi server dengan antrian Amazon SQS
- Mengkonfigurasi tag untuk antrian Amazon SQS
- Mengirim atribut pesan ke antrian Amazon SQS

Menggunakan enkripsi sisi server dengan antrian Amazon SQS

Anda dapat menggunakan AWS SDK for Java untuk menambahkan enkripsi sisi server (SSE) ke antrian Amazon SQS. Setiap antrian menggunakan kunci AWS Key Management Service (AWS KMS) KMS untuk menghasilkan kunci enkripsi data. Contoh ini menggunakan kunci KMS AWS terkelola untuk Amazon SQS. Untuk informasi selengkapnya tentang penggunaan SSE dan peran kunci KMS, lihat. Enkripsi saat istirahat di Amazon SQS

Menambahkan SSE ke antrian yang ada

Untuk mengaktifkan enkripsi sisi server untuk antrian yang ada, gunakan `SetQueueAttributes` metode untuk mengatur atribut. `KmsMasterKeyId`

Contoh kode berikut menetapkan AWS KMS key sebagai kunci KMS AWS terkelola untuk Amazon SQS. Contoh ini juga mengatur periode AWS KMS key penggunaan kembali menjadi 140 detik.

Sebelum Anda menjalankan kode contoh, pastikan Anda telah menetapkan AWS kredensialnya. Untuk informasi selengkapnya, lihat Menyiapkan AWS Kredensial dan Wilayah untuk Pengembangan di Panduan AWS SDK for Java 2.x Pengembang.

```
// 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);
```

Menonaktifkan SSE untuk antrian

Untuk menonaktifkan enkripsi sisi server untuk antrian yang ada, atur `KmsMasterKeyId` atribut ke string kosong menggunakan metode. `SetQueueAttributes`

> **Important**
>
> `null` bukanlah nilai yang valid untuk `KmsMasterKeyId`.

Membuat antrian dengan SSE

Untuk mengaktifkan SSE saat Anda membuat antrian, tambahkan `KmsMasterKeyId` atribut ke metode `CreateQueue` API.

Contoh berikut membuat antrian baru dengan SSE diaktifkan. Antrian menggunakan kunci KMS AWS terkelola untuk Amazon SQS. Contoh ini juga mengatur periode AWS KMS key penggunaan kembali menjadi 160 detik.

Sebelum Anda menjalankan kode contoh, pastikan Anda telah menetapkan AWS kredensialnya. Untuk informasi selengkapnya, lihat Menyiapkan AWS Kredensial dan Wilayah untuk Pengembangan di Panduan AWS SDK for Java 2.x Pengembang.

```
// 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);
```

Mengambil atribut SSE

Untuk informasi tentang mengambil atribut antrian, lihat Contoh di Referensi API Layanan Antrian Sederhana Amazon.

Untuk mengambil ID kunci KMS atau periode penggunaan kembali kunci data untuk antrian tertentu, jalankan `GetQueueAttributes` metode dan ambil nilai dan. `KmsMasterKeyId` `KmsDataKeyReusePeriodSeconds`

Mengkonfigurasi tag untuk antrian Amazon SQS

Gunakan tag alokasi biaya untuk membantu mengatur dan mengidentifikasi antrian Amazon SQS Anda. Contoh berikut menunjukkan cara mengkonfigurasi tag menggunakan AWS SDK for Java. Untuk informasi selengkapnya, lihat Tag alokasi biaya Amazon SQS.

Sebelum Anda menjalankan kode contoh, pastikan Anda telah menetapkan AWS kredensialnya. Untuk informasi selengkapnya, lihat Menyiapkan AWS Kredensial dan Wilayah untuk Pengembangan di Panduan AWS SDK for Java 2.x Pengembang.

Mencantumkan tanda

Untuk membuat daftar tag untuk antrian, gunakan `ListQueueTags` metode ini.

```
// 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() ));
```

Menambahkan atau memperbarui tag

Untuk menambahkan atau memperbarui nilai tag untuk antrian, gunakan `TagQueue` metode ini.

```
 // 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);
```

Menghapus tanda

Untuk menghapus satu atau beberapa tag dari antrian, gunakan `UntagQueue` metode ini. Contoh berikut menghapus `Accounting ID` tag.

```

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

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

Mengirim atribut pesan ke antrian Amazon SQS

Anda dapat menyertakan metadata terstruktur (seperti stempel waktu, data geospasial, tanda tangan, dan pengenal) dengan pesan menggunakan atribut pesan. Untuk informasi selengkapnya, lihat Atribut pesan Amazon SQS.

Sebelum Anda menjalankan kode contoh, pastikan Anda telah menetapkan AWS kredensialnya. Untuk informasi selengkapnya, lihat Mengatur AWS Kredensial dan Wilayah untuk Pengembangan di Panduan AWS SDK for Java 2.x Pengembang.

Mendefinisikan atribut

Untuk menentukan atribut pesan, tambahkan kode berikut, yang menggunakan tipe `MessageAttributeValue` data. Untuk informasi selengkapnya, lihat Komponen atribut pesan dan Tipe data atribut pesan.

AWS SDK for Java Secara otomatis menghitung checksum isi pesan dan atribut pesan dan membandingkannya dengan data yang dikembalikan Amazon SQS. Untuk informasi selengkapnya, lihat Panduan AWS SDK for Java 2.x Pengembang dan Menghitung intisari pesan MD5 untuk atribut pesan untuk bahasa pemrograman lainnya.

String

Contoh ini mendefinisikan `String` atribut bernama `Name` dengan nilai`Jane`.

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

Number

Contoh ini mendefinisikan `Number` atribut bernama `AccurateWeight` dengan nilai`230.00000000000000001`.

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

Binary

Contoh ini mendefinisikan `Binary` atribut bernama `ByteArray` dengan nilai array 10-byte yang tidak diinisialisasi.

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

String (custom)

Contoh ini mendefinisikan atribut kustom `String.EmployeeId` bernama `EmployeeId` dengan nilai`ABC123456`.

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

Number (custom)

Contoh ini mendefinisikan atribut kustom `Number.AccountId` bernama `AccountId` dengan nilai`000123456`.

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

> **Note**
>
> Karena tipe data dasar adalah`Number`, `ReceiveMessage` metode kembali`123456`.

Binary (custom)

Contoh ini mendefinisikan atribut kustom `Binary.JPEG` bernama `ApplicationIcon` dengan nilai array 10-byte yang tidak diinisialisasi.

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

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

Mengirim pesan dengan atribut

Contoh ini menambahkan atribut ke `SendMessageRequest` sebelum mengirim pesan.

```
// 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

Jika Anda mengirim pesan ke antrean First-In-First-Out (FIFO), pastikan `sendMessage` metode dijalankan setelah Anda memberikan ID grup pesan.
Jika Anda menggunakan `SendMessageBatch` metode ini`SendMessage`, Anda harus menentukan atribut pesan untuk setiap pesan dalam batch.

Bekerja dengan Amazon SQS API

Bagian ini memberikan informasi tentang membangun endpoint Amazon SQS, membuat permintaan API kueri menggunakan metode GET dan POST, dan menggunakan tindakan API batch. Untuk informasi terperinci tentang tindakan Amazon SQS —termasuk parameter, kesalahan, contoh, dan tipe data, lihat Referensi API Layanan Antrian Sederhana Amazon.

Untuk mengakses Amazon SQS menggunakan berbagai bahasa pemrograman, Anda juga dapat menggunakan AWS SDK, yang berisi fungsionalitas otomatis berikut:

- Secara kriptografi menandatangani permintaan layanan Anda
- Mencoba kembali permintaan
- Menangani respons kesalahan

Untuk informasi alat baris perintah, lihat bagian Amazon SQS di Referensi AWS CLI Perintah dan Referensi AWS Tools for PowerShell Cmdlet.

Amazon SQS API dengan AWS protokol JSON

Amazon SQS menggunakan protokol AWS JSON sebagai mekanisme transportasi untuk semua Amazon SQS API pada versi SDK yang ditentukan.AWS AWS Protokol JSON menyediakan throughput yang lebih tinggi, latensi yang lebih rendah, dan komunikasi yang lebih cepat. application-to-application AWS Protokol JSON lebih efisien dalam serialisasi/deserialisasi permintaan dan tanggapan jika dibandingkan dengan protokol kueri. AWS Jika Anda masih lebih suka menggunakan protokol AWS kueri dengan SQS API, lihat Bahasa apa yang didukung untuk protokol AWS JSON yang digunakan di Amazon SQS API? versi AWS SDK yang mendukung protokol kueri Amazon AWS SQS.

Amazon SQS menggunakan protokol AWS JSON untuk berkomunikasi antara klien AWS SDK (misalnya, Java, Python, Golang,) dan JavaScript server Amazon SQS. Permintaan HTTP dari operasi Amazon SQS API menerima input berformat JSON. Operasi Amazon SQS dijalankan, dan respons eksekusi dikirim kembali ke klien SDK dalam format JSON. Dibandingkan dengan AWS query, AWS JSON lebih sederhana, lebih cepat, dan lebih efisien untuk mengangkut data antara klien dan server.

- AWS Protokol JSON bertindak sebagai mediator antara klien dan server Amazon SQS.

- Server tidak memahami bahasa pemrograman di mana operasi Amazon SQS dibuat, tetapi memahami protokol AWS JSON.
- Protokol AWS JSON menggunakan serialisasi (konversi objek ke format JSON) dan de-serialisasi (konversi format JSON ke objek) antara klien Amazon SQS dan server.

Untuk informasi selengkapnya tentang protokol AWS JSON dengan Amazon SQS, lihat. FAQ protokol Amazon SQS AWS JSON

AWS Protokol JSON tersedia pada versi AWS SDK yang ditentukan. Untuk meninjau versi SDK dan tanggal rilis di seluruh varian bahasa, lihat AWS matriks dukungan versi SDK dan Alat di Panduan Referensi AWS SDK dan Alat

Topik

Membuat permintaan API kueri menggunakan protokol AWS JSON di Amazon SQS

Di bagian ini Anda mempelajari cara membuat endpoint Amazon SQS, membuat permintaan POST, dan menafsirkan tanggapan.

> **Note**
>
> AWS Protokol JSON didukung untuk sebagian besar varian bahasa. Untuk daftar lengkap varian bahasa yang didukung, lihatBahasa apa yang didukung untuk protokol AWS JSON yang digunakan di Amazon SQS API?.

Topik

- Menafsirkan tanggapan Amazon SQS JSON API
- FAQ protokol Amazon SQS AWS JSON

Membangun titik akhir

Untuk bekerja dengan antrian Amazon SQS, Anda harus membuat titik akhir. Untuk informasi tentang titik akhir Amazon SQS, lihat halaman berikut di: Referensi Umum Amazon Web

- Titik akhir regional
- Titik akhir dan kuota Layanan Antrian Sederhana Amazon

Setiap titik akhir Amazon SQS bersifat independen. Misalnya, jika dua antrian diberi nama MyQueuedan satu memiliki titik akhir `sqs.us-east-2.amazonaws.com` sementara yang lain memiliki titik akhir`sqs.eu-west-2.amazonaws.com`, kedua antrian tidak berbagi data apa pun satu sama lain.

Berikut ini adalah contoh dari endpoint yang membuat permintaan untuk membuat antrian.

```
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**
>
> Nama antrian dan URL antrian peka huruf besar/kecil.

Struktur *AUTHPARAMS*tergantung pada tanda tangan permintaan API. Untuk informasi selengkapnya, lihat Menandatangani Permintaan AWS API di Referensi Umum Amazon Web Services.

Membuat permintaan POST

Permintaan Amazon SQS POST mengirimkan parameter kueri sebagai formulir di badan permintaan HTTP.

Berikut ini adalah contoh dari header HTTP dengan `X-Amz-Target` set ke`AmazonSQS.<operationName>`, dan header HTTP dengan `Content-Type` set ke`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",
}
```

Permintaan HTTP POST ini mengirimkan pesan ke antrian Amazon SQS.

> **Note**
>
> Baik header HTTP `X-Amz-Target` dan `Content-Type` diperlukan.
> Klien HTTP Anda mungkin menambahkan item lain ke permintaan HTTP, sesuai dengan versi HTTP klien.

Menafsirkan tanggapan Amazon SQS JSON API

Menanggapi permintaan tindakan, Amazon SQS mengembalikan struktur data JSON yang berisi hasil permintaan. Untuk informasi selengkapnya, lihat tindakan individual di Referensi API Layanan Antrian Sederhana Amazon danFAQ protokol Amazon SQS AWS JSON.

Topik

Struktur respons JSON yang sukses

Jika permintaan berhasil, elemen respons utama adalah`x-amzn-RequestId`, yang berisi Universal Unique Identifier (UUID) permintaan, serta bidang respons tambahan lainnya. Misalnya, `CreateQueue` respons berikut berisi `QueueUrl` bidang, yang, pada gilirannya, berisi URL antrian yang dibuat.

```
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"
}
```

Struktur respons kesalahan JSON

Jika permintaan tidak berhasil, Amazon SQS mengembalikan respons utama, termasuk header HTTP dan isi.

Di header HTTP, `x-amzn-RequestId` berisi UUID permintaan. `x-amzn-query-error`berisi dua informasi: jenis kesalahan, dan apakah kesalahan itu adalah kesalahan produsen atau konsumen.

Di badan respons, `"__type"` menunjukkan detail kesalahan lainnya, dan `Message` menunjukkan kondisi kesalahan dalam format yang dapat dibaca.

Berikut ini adalah contoh respon kesalahan dalam format 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."
}
```

FAQ protokol Amazon SQS AWS JSON

Pertanyaan yang sering diajukan tentang penggunaan protokol AWS JSON dengan Amazon SQS.

Apa itu protokol AWS JSON, dan apa bedanya dengan permintaan dan tanggapan Amazon SQS API yang ada?

JSON adalah salah satu metode pengkabelan yang paling banyak digunakan dan diterima untuk komunikasi antara sistem heterogen. Amazon SQS menggunakan JSON sebagai media untuk berkomunikasi antara klien AWS SDK (misalnya, Java, Python, Golang,) JavaScript dan server Amazon SQS. Permintaan HTTP dari operasi Amazon SQS API menerima masukan dalam bentuk JSON. Operasi Amazon SQS dijalankan dan respons eksekusi dibagikan kembali ke klien SDK dalam bentuk JSON. Dibandingkan dengan AWS query, JSON lebih efisien dalam mengangkut data antara klien dan server.

- Protokol Amazon SQS AWS JSON bertindak sebagai mediator antara klien dan server Amazon SQS.
- Server tidak memahami bahasa pemrograman di mana operasi Amazon SQS dibuat, tetapi memahami protokol AWS JSON.
- Protokol Amazon SQS AWS JSON menggunakan serialisasi (konversi objek ke format JSON) dan deserialisasi (konversi format JSON ke objek) antara klien Amazon SQS dan server.

Bagaimana cara memulai dengan protokol AWS JSON untuk Amazon SQS?

Untuk memulai dengan versi AWS SDK terbaru untuk mencapai pengiriman pesan yang lebih cepat untuk Amazon SQS, tingkatkan SDK AWS Anda ke versi yang ditentukan atau versi berikutnya. Untuk mempelajari lebih lanjut tentang klien SDK, lihat kolom Panduan pada tabel di bawah ini.

Berikut ini adalah daftar versi SDK di seluruh varian bahasa untuk protokol AWS JSON untuk digunakan dengan Amazon SQS API:

Bahasa	Repositori klien SDK	Versi klien SDK yang diperlukan	Panduan
C++	aws/aws-sdk-cpp	1.11.98	AWS SDK for C++
Golang 1.x	aws/aws-sdk-go	v1.47.7	AWS SDK for Go
Golang 2.x	aws/aws-sdk-go-v2	v1.28.0	AWS SDK for Go V2
Java 1.x	aws/aws-sdk-java	1.12.585	AWS SDK for Java
Java 2.x	aws/aws-sdk-java-v2	2.21.19	AWS SDK for Java
JavaScript v2.x	aws/aws-sdk-js	v2.1492.0	JavaScript pada AWS
JavaScript v3.x	aws/aws-sdk-js-v3	v3.447.0	JavaScript pada AWS
.NET	aws/aws-sdk-net	3.7.681.0	AWS SDK for .NET
PHP	aws/aws-sdk-php	3.285.2	AWS SDK for PHP
Python-Boto3	boto/boto3	1.28.82	AWS SDK untuk Python (Boto3)
Python-botocore	boto/botocore	1.31.82	AWS SDK untuk Python (Boto3)
awscli	AWS CLI	1.29.82	AWS Antarmuka Baris Perintah

Bahasa	Repositori klien SDK	Versi klien SDK yang diperlukan	Panduan
Ruby	aws/aws-sdk-ruby	1.67.0	AWS SDK for Ruby

Apa risiko mengaktifkan protokol JSON untuk beban kerja Amazon SQS saya?

Jika Anda menggunakan implementasi khusus AWS SDK atau kombinasi klien kustom dan AWS SDK untuk berinteraksi dengan Amazon SQS yang AWS menghasilkan respons berbasis Kueri (alias berbasis XML), mungkin tidak kompatibel dengan protokol JSON. AWS Jika Anda mengalami masalah, hubungi AWS Support.

Bagaimana jika saya sudah menggunakan versi AWS SDK terbaru, tetapi solusi open source saya tidak mendukung JSON?

Anda harus mengubah versi SDK Anda ke versi sebelumnya yang Anda gunakan. Lihat Bagaimana cara memulai dengan protokol AWS JSON untuk Amazon SQS? untuk informasi lebih lanjut. AWS Versi SDK yang tercantum dalam Bagaimana cara memulai dengan protokol AWS JSON untuk Amazon SQS? menggunakan protokol kawat JSON untuk Amazon SQS API. Jika Anda mengubah AWS SDK ke versi sebelumnya, API Amazon SQS Anda akan menggunakan AWS kueri.

Bahasa apa yang didukung untuk protokol AWS JSON yang digunakan di Amazon SQS API?

Amazon SQS mendukung semua varian bahasa di mana AWS SDK umumnya tersedia (GA). Saat ini, kami tidak mendukung Kotlin, Rust, atau Swift. Untuk mempelajari lebih lanjut tentang varian bahasa lain, lihat Alat untuk Dibangun AWS.

Wilayah apa yang didukung untuk protokol AWS JSON yang digunakan di Amazon SQS API

Amazon SQS mendukung protokol AWS JSON di semua AWS wilayah tempat Amazon SQS tersedia.

Peningkatan latensi apa yang dapat saya harapkan saat memutakhirkan ke versi AWS SDK yang ditentukan untuk Amazon SQS menggunakan protokol JSON? AWS

AWS Protokol JSON lebih efisien dalam serialisasi dan deserialisasi permintaan dan tanggapan jika dibandingkan dengan protokol kueri. AWS Berdasarkan pengujian AWS kinerja untuk muatan pesan 5 KB, protokol JSON untuk Amazon SQS end-to-end mengurangi latensi pemrosesan pesan hingga 23%, dan mengurangi penggunaan CPU dan memori sisi klien aplikasi.

Apakah protokol AWS kueri akan tidak digunakan lagi?

AWS protokol kueri akan terus didukung. Anda dapat terus menggunakan protokol AWS kueri selama versi AWS SDK Anda disetel versi sebelumnya selain yang tercantum di Bagaimana cara memulai dengan protokol AWS JSON untuk Amazon SQS.

Di mana saya dapat menemukan informasi lebih lanjut tentang protokol AWS JSON?

Anda dapat menemukan informasi lebih lanjut tentang protokol JSON di protokol AWS JSON 1.0 di dokumentasi Smithy. Untuk selengkapnya tentang permintaan Amazon SQS API menggunakan protokol AWS JSON, lihat. Membuat permintaan API kueri menggunakan protokol AWS JSON di Amazon SQS

Membuat permintaan API kueri menggunakan protokol AWS kueri di Amazon SQS

Di bagian ini Anda mempelajari cara membuat endpoint Amazon SQS, membuat permintaan GET dan POST, dan menafsirkan tanggapan.

Topik

- Membangun titik akhir
- Membuat permintaan GET
- Membuat permintaan POST
- Menafsirkan tanggapan Amazon SQS XMLAPI

Membangun titik akhir

Agar dapat bekerja dengan antrian Amazon SQS, Anda harus membuat titik akhir. Untuk informasi tentang titik akhir Amazon SQS, lihat halaman berikut di: Referensi Umum Amazon Web

- Titik akhir regional
- Titik akhir dan kuota Layanan Antrian Sederhana Amazon

Setiap titik akhir Amazon SQS bersifat independen. Misalnya, jika dua antrian diberi nama MyQueuedan satu memiliki titik akhir `sqs.us-east-2.amazonaws.com` sementara yang lain memiliki titik akhir`sqs.eu-west-2.amazonaws.com`, kedua antrian tidak berbagi data apa pun satu sama lain.

Berikut ini adalah contoh dari endpoint yang membuat permintaan untuk membuat antrian.

```
https://sqs.eu-west-2.amazonaws.com/
?Action=CreateQueue
&DefaultVisibilityTimeout=40
&QueueName=MyQueue
&Version=2012-11-05
&AUTHPARAMS
```

> **Note**
>
> Nama antrian dan URL antrian peka huruf besar/kecil.
> Struktur *AUTHPARAMS*tergantung pada tanda tangan permintaan API. Untuk informasi selengkapnya, lihat Menandatangani Permintaan AWS API di Referensi Umum Amazon Web Services.

Membuat permintaan GET

Permintaan Amazon SQS GET disusun sebagai URL yang terdiri dari berikut ini:

- Endpoint — Sumber daya tempat permintaan bertindak (nama antrian dan URL), misalnya: `https://sqs.us-east-2.amazonaws.com/`*`123456789012`*`/MyQueue`
- Tindakan — Tindakan yang ingin Anda lakukan di titik akhir. Tanda tanya (?) memisahkan titik akhir dari tindakan, misalnya: `?Action=SendMessage&MessageBody=Your%20Message%20Text`
- Parameter — Parameter permintaan apa pun. Setiap parameter dipisahkan oleh ampersand (&), misalnya: `&Version=2012-11-05&`*`AUTHPARAMS`*

Berikut ini adalah contoh permintaan GET yang mengirim pesan ke antrian Amazon SQS.

```
https://sqs.us-east-2.amazonaws.com/123456789012/MyQueue
?Action=SendMessage&MessageBody=Your%20message%20text
&Version=2012-11-05
&AUTHPARAMS
```

> **Note**
>
> Nama antrian dan URL antrian peka huruf besar/kecil.
> Karena permintaan GET adalah URL, Anda harus menyandikan URL semua nilai parameter. Karena spasi tidak diizinkan di URL, setiap spasi dikodekan URL sebagai. `%20` Contoh lainnya tidak dikodekan URL untuk membuatnya lebih mudah dibaca.

Membuat permintaan POST

Permintaan Amazon SQS POST mengirimkan parameter kueri sebagai formulir di badan permintaan HTTP.

Berikut ini adalah contoh header HTTP dengan `Content-Type` set `keapplication/x-www-form-urlencoded`.

```
POST /123456789012/MyQueue HTTP/1.1
Host: sqs.us-east-2.amazonaws.com
Content-Type: application/x-www-form-urlencoded
```

Header diikuti oleh permintaan `form-urlencoded` GET yang mengirim pesan ke antrian Amazon SQS. Setiap parameter dipisahkan oleh ampersand ()&.

```
Action=SendMessage
&MessageBody=Your+Message+Text
&Expires=2020-10-15T12%3A00%3A00Z
&Version=2012-11-05
&AUTHPARAMS
```

> **Note**
>
> Hanya header `Content-Type` HTTP yang diperlukan. Sama seperti untuk permintaan GET. *AUTHPARAMS*

Klien HTTP Anda mungkin menambahkan item lain ke permintaan HTTP, sesuai dengan versi HTTP klien.

Menafsirkan tanggapan Amazon SQS XMLAPI

Menanggapi permintaan tindakan, Amazon SQS mengembalikan struktur data XMLyang berisi hasil permintaan. Untuk informasi selengkapnya, lihat tindakan individual di Referensi API Layanan Antrian Sederhana Amazon.

Topik

Struktur respons XHTML yang sukses

Jika permintaan berhasil, elemen respons utama dinamai setelah tindakan, dengan `Response` ditambahkan (misalnya,*`ActionName`*`Response`).

Elemen ini berisi elemen turunan berikut:

- ***`ActionName`*`Result`**— Berisi elemen khusus tindakan. Misalnya, `CreateQueueResult` elemen berisi `QueueUrl` elemen yang, pada gilirannya, berisi URL antrian yang dibuat.
- **`ResponseMetadata`**— Berisi `RequestId` yang, pada gilirannya, berisi Universal Unique Identifier (UUID) dari permintaan.

Berikut ini adalah contoh respon sukses dalam format 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>
```

Struktur respon kesalahan XML

Jika permintaan tidak berhasil, Amazon SQS selalu mengembalikan elemen respons utama. `ErrorResponse` Elemen ini berisi elemen `Error` dan elemen `RequestId`.

`Error`Elemen berisi elemen anak berikut:

- **`Type`**- Menentukan apakah kesalahan adalah produsen atau kesalahan konsumen.
- **`Code`**- Menentukan jenis kesalahan.
- **`Message`**- Menentukan kondisi kesalahan dalam format yang dapat dibaca.
- **`Detail`**- (Opsional) Menentukan rincian tambahan tentang kesalahan.

`RequestId`Elemen berisi UUID permintaan.

Berikut ini adalah contoh respon kesalahan dalam format 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>
```

Mengautentikasi permintaan untuk Amazon SQS

Otentikasi adalah proses mengidentifikasi dan memverifikasi pihak yang mengirim permintaan. Selama tahap pertama otentikasi, AWS verifikasi identitas produsen dan apakah produsen terdaftar untuk digunakan AWS(untuk informasi lebih lanjut, lihatLangkah 1: Buat pengguna Akun AWS dan IAM). Selanjutnya, AWS patuhi prosedur berikut:

1. Produser (pengirim) memperoleh kredensi yang diperlukan.

2. Produser mengirimkan permintaan dan kredensi kepada konsumen (penerima).
3. Konsumen menggunakan kredensi untuk memverifikasi apakah produsen mengirim permintaan.
4. Salah satu hal berikut terjadi:
   - Jika otentikasi berhasil, konsumen memproses permintaan.
   - Jika otentikasi gagal, konsumen menolak permintaan dan mengembalikan kesalahan.

Topik

Proses otentikasi dasar dengan HMAC-SHA

Saat mengakses Amazon SQS menggunakan Query API, Anda harus memberikan item berikut untuk mengautentikasi permintaan Anda:

- ID Kunci AWS Akses yang mengidentifikasi Anda Akun AWS, yang AWS digunakan untuk mencari Kunci Akses Rahasia Anda.
- Tanda tangan permintaan HMAC-SHA, dihitung menggunakan Kunci Akses Rahasia Anda (rahasia bersama yang hanya diketahui oleh Anda dan AWS—untuk informasi selengkapnya, lihat RFC2104). AWS SDK menangani proses penandatanganan; namun, jika Anda mengirimkan permintaan kueri melalui HTTP atau HTTPS, Anda harus menyertakan tanda tangan di setiap permintaan kueri.

  1. Dapatkan Kunci Penandatanganan Versi Tanda Tangan 4. Untuk informasi selengkapnya, lihat Menurunkan Kunci Penandatanganan dengan Java.

     > **Note**
     >
     > Amazon SQS mendukung Signature Version 4, yang memberikan peningkatan keamanan dan kinerja berbasis SHA256 dibandingkan versi sebelumnya. Saat Anda membuat aplikasi baru yang menggunakan Amazon SQS, gunakan Signature Version 4.

  2. Base64-encode tanda tangan permintaan. Contoh kode Java berikut melakukan ini:

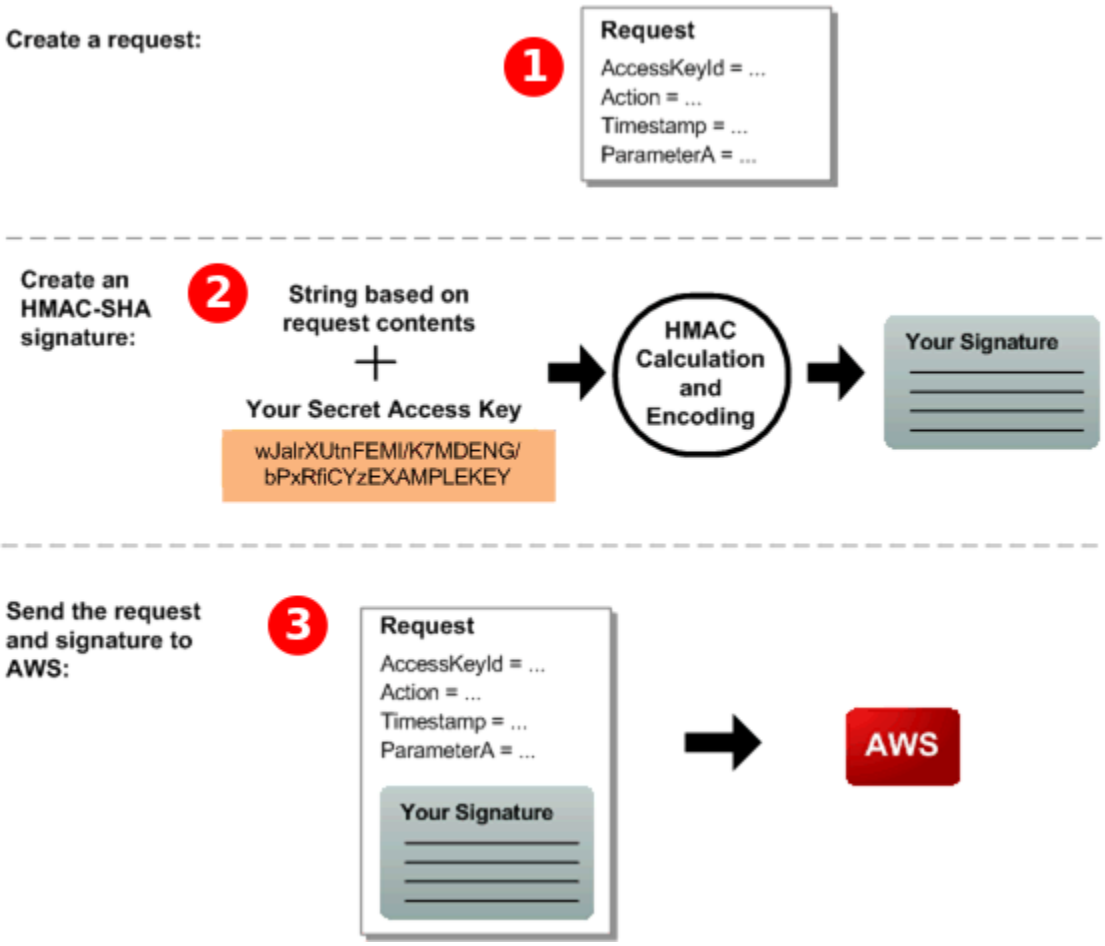
```
package amazon.webservices.common;

// 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);
    }
}
```

- Stempel waktu (atau kedaluwarsa) permintaan. Stempel waktu yang Anda gunakan dalam permintaan harus berupa `dateTime` objek, dengan tanggal lengkap, termasuk jam, menit, dan detik. Misalnya: `2007-01-31T23:59:59Z` Meskipun ini tidak diperlukan, kami sarankan untuk menyediakan objek menggunakan zona waktu Waktu Universal Terkoordinasi (Greenwich Mean Time).

> **Note**
>
> Pastikan waktu server Anda diatur dengan benar. Jika Anda menentukan stempel waktu (bukan kedaluwarsa), permintaan secara otomatis akan kedaluwarsa 15 menit setelah waktu yang ditentukan (AWS tidak memproses permintaan dengan stempel waktu lebih dari 15 menit lebih awal dari waktu saat ini di server). AWS
> Jika Anda menggunakan .NET, Anda tidak boleh mengirim stempel waktu yang terlalu spesifik (karena interpretasi yang berbeda tentang bagaimana presisi waktu ekstra harus dihapus). Dalam hal ini, Anda harus secara manual membangun `dateTime` objek dengan presisi tidak lebih dari satu milidetik.

Bagian 1: Permintaan dari pengguna

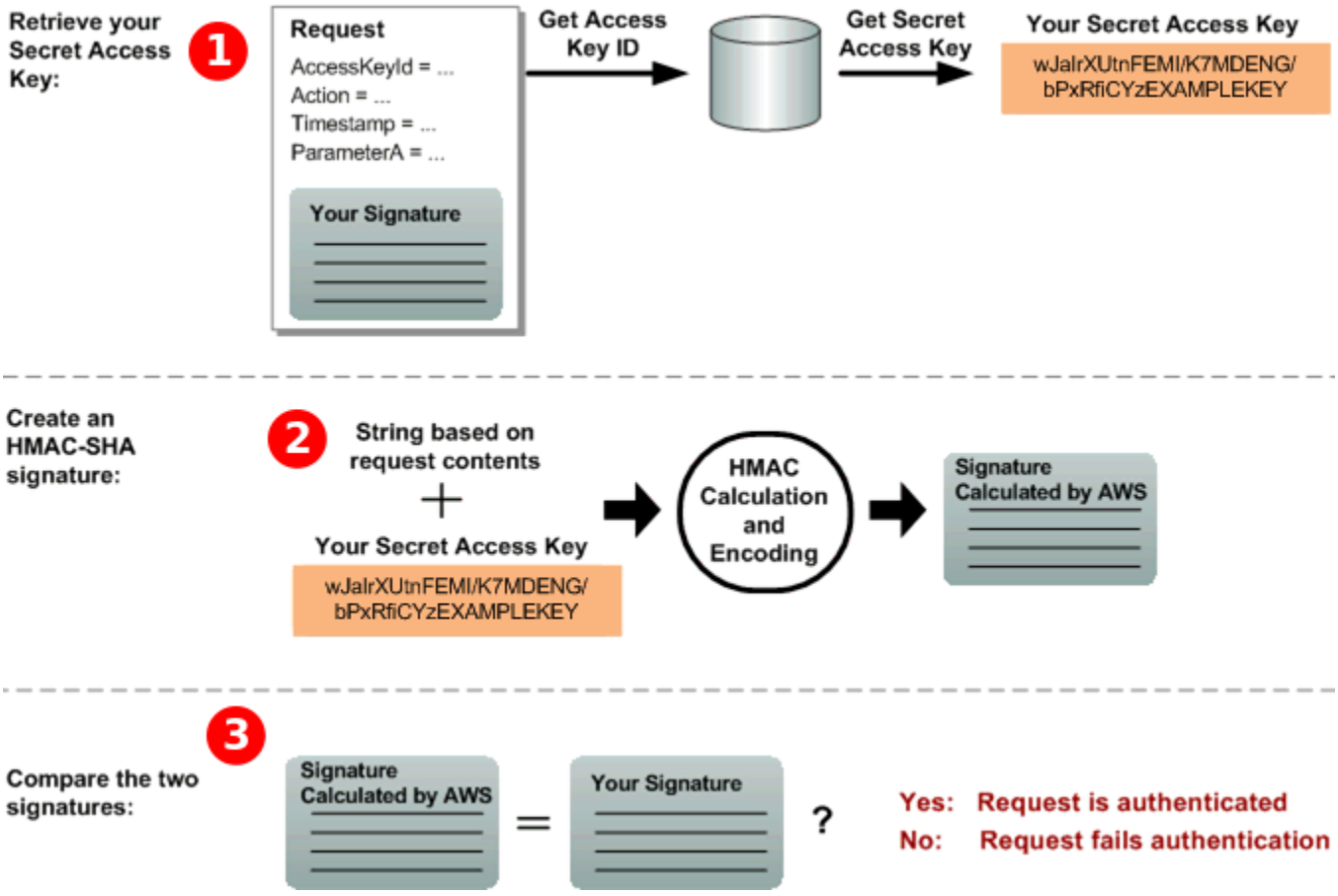
Berikut ini adalah proses yang harus Anda ikuti untuk mengautentikasi AWS permintaan menggunakan tanda tangan permintaan HMAC-SHA.



1. Membangun permintaan untuk AWS.
2. Hitung tanda tangan kode otentikasi pesan hash kunci (HMAC-SHA) menggunakan Kunci Akses Rahasia Anda.
3. Sertakan tanda tangan dan ID Kunci Akses Anda dalam permintaan, lalu kirim permintaan ke AWS.

Bagian 2: Tanggapan dari AWS

AWS memulai proses berikut sebagai tanggapan.



1. AWS menggunakan ID Kunci Akses untuk mencari Kunci Akses Rahasia Anda.
2. AWS menghasilkan tanda tangan dari data permintaan dan Kunci Akses Rahasia, menggunakan algoritma yang sama yang Anda gunakan untuk menghitung tanda tangan yang Anda kirim dalam permintaan.
3. Salah satu hal berikut terjadi:
   - Jika tanda tangan yang AWS dihasilkan cocok dengan yang Anda kirim dalam permintaan, AWS anggap permintaan tersebut asli.
   - Jika perbandingan gagal, permintaan dibuang, dan AWS mengembalikan kesalahan.

Tindakan batch Amazon SQS

Untuk mengurangi biaya atau memanipulasi hingga 10 pesan dengan satu tindakan, Anda dapat menggunakan tindakan berikut:

- `SendMessageBatch`

- `DeleteMessageBatch`
- `ChangeMessageVisibilityBatch`

Anda dapat memanfaatkan fungsionalitas batch menggunakan Query API, atau AWS SDK yang mendukung tindakan batch Amazon SQS.

> **Note**
>
> Ukuran total semua pesan yang Anda kirim dalam satu `SendMessageBatch` panggilan tidak dapat melebihi 262.144 byte (256 KiB).
> Anda tidak dapat mengatur izin untuk`SendMessageBatch`,`DeleteMessageBatch`, atau `ChangeMessageVisibilityBatch` secara eksplisit. Menyetel izin untuk `SendMessageDeleteMessage`,, atau `ChangeMessageVisibility` menetapkan izin untuk versi batch tindakan yang sesuai.
> Konsol Amazon SQS tidak mendukung tindakan batch.

Topik

- Mengaktifkan buffering sisi klien dan batching permintaan dengan Amazon SQS
- Meningkatkan throughput menggunakan penskalaan horizontal dan batching aksi dengan Amazon SQS

Mengaktifkan buffering sisi klien dan batching permintaan dengan Amazon SQS

AWS SDK for JavaTermasuk `AmazonSQSBufferedAsyncClient` yang mengakses Amazon SQS. Klien ini memungkinkan pengelompokan permintaan sederhana menggunakan buffering sisi klien—panggilan yang dilakukan dari klien pertama kali di-buffer dan kemudian dikirim sebagai permintaan batch ke Amazon SQS.

Buffering sisi klien memungkinkan hingga 10 permintaan untuk di-buffer dan dikirim sebagai permintaan batch, mengurangi biaya penggunaan Amazon SQS dan mengurangi jumlah permintaan yang dikirim. `AmazonSQSBufferedAsyncClient`buffer panggilan sinkron dan asinkron. Permintaan batch dan dukungan untuk polling panjang juga dapat membantu meningkatkan throughput. Untuk informasi selengkapnya, lihat Meningkatkan throughput menggunakan penskalaan horizontal dan batching aksi dengan Amazon SQS.

Karena `AmazonSQSBufferedAsyncClient` mengimplementasikan antarmuka yang sama seperti`AmazonSQSAsyncClient`, migrasi dari `AmazonSQSAsyncClient` ke `AmazonSQSBufferedAsyncClient` biasanya hanya memerlukan sedikit perubahan pada kode Anda yang ada.

> **Note**
>
> Klien Asinkron Buffered Amazon SQS saat ini tidak mendukung antrian FIFO.

Topik

- Menggunakan Klien AmazonSQS BufferedAsync
- Mengkonfigurasi Klien BufferedAsync AmazonSQS

Menggunakan Klien AmazonSQS BufferedAsync

Sebelum memulai, selesaikan langkah-langkah di Menyiapkan Amazon SQS.

> **Important**
>
> Saat ini AWS SDK for Java 2.x tidak kompatibel dengan`AmazonSQSBufferedAsyncClient`.

Anda dapat membuat yang baru `AmazonSQSBufferedAsyncClient` berdasarkan`AmazonSQSAsyncClient`, misalnya:

```
// Create the basic Amazon SQS async client
final AmazonSQSAsync sqsAsync = new AmazonSQSAsyncClient();

// Create the buffered client
final AmazonSQSAsync bufferedSqs = new AmazonSQSBufferedAsyncClient(sqsAsync);
```

Setelah Anda membuat yang baru`AmazonSQSBufferedAsyncClient`, Anda dapat menggunakannya untuk mengirim beberapa permintaan ke Amazon SQS (seperti yang Anda bisa dengan`AmazonSQSAsyncClient`), misalnya:

```
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);
```

Mengkonfigurasi Klien BufferedAsync AmazonSQS

`AmazonSQSBufferedAsyncClient`telah dikonfigurasi sebelumnya dengan pengaturan yang berfungsi untuk sebagian besar kasus penggunaan. Anda dapat mengkonfigurasi lebih lanjut`AmazonSQSBufferedAsyncClient`, misalnya:

1. Buat instance `QueueBufferConfig` kelas dengan parameter konfigurasi yang diperlukan.
2. Berikan instance ke `AmazonSQSBufferedAsyncClient` konstruktor.

```
// 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);
```

QueueBufferConfig parameter konfigurasi

Parameter	Nilai default	Deskripsi
`longPoll`	`true`	Ketika `longPoll` diatur ke`true`, `AmazonSQS`

Parameter	Nilai default	Deskripsi
		`BufferedAsyncClient` mencoba untuk menggunakan polling panjang ketika mengkonsumsi pesan.
`longPollWaitTimeoutSeconds`	20 s	Jumlah waktu maksimum (dalam detik) yang diblokir `ReceiveMessage` panggilan di server, menunggu pesan muncul dalam antrian sebelum kembali dengan hasil penerimaan kosong. **Note** Ketika polling panjang dinonaktifkan, pengaturan ini tidak berpengaruh.

Parameter	Nilai default	Deskripsi
`maxBatchOpenMs`	200 ms	Jumlah waktu maksimum (dalam milidetik) panggilan keluar menunggu panggilan lain yang dengannya ia mengumpulkan pesan dari jenis yang sama. Semakin tinggi pengaturan, semakin sedikit batch yang diperlukan untuk melakukan jumlah pekerjaan yang sama (namun, panggilan pertama dalam batch harus menghabiskan waktu lebih lama menunggu). Saat Anda menyetel parameter ini`0`, permintaa n yang dikirimkan tidak menunggu permintaan lain, yang secara efektif menonakti fkan batching.

Parameter	Nilai default	Deskripsi
`maxBatchSize`	10 permintaan per batch	Jumlah maksimum pesan yang dikumpulkan bersama dalam satu permintaan. Semakin tinggi pengaturan, semakin sedikit batch yang diperlukan untuk melakukan jumlah permintaan yang sama. **Note** 10 permintaan per batch adalah nilai maksimum yang diizinkan untuk Amazon SQS.
`maxBatchSizeBytes`	256 KiB	Ukuran maksimum kumpulan pesan, dalam byte, yang coba dikirim klien ke Amazon SQS. **Note** 256 KiB adalah nilai maksimum yang diizinkan untuk Amazon SQS.

Parameter	Nilai default	Deskripsi
`maxDoneReceiveBatches`	10 batch	Jumlah maksimum batch penerima yang `AmazonSQSBufferedAsyncClient` mengambil dan menyimpan sisi klien. Semakin tinggi pengaturan, semakin banyak permintaan penerimaan yang dapat dipenuhi tanpa harus melakukan panggilan ke Amazon SQS (namun, semakin banyak pesan yang diambil sebelumnya, semakin lama mereka tetap berada di buffer, menyebabkan batas waktu visibilitas mereka sendiri kedaluwarsa). **Note** `0`menunjukkan bahwa semua pra-pengambilan pesan dinonaktifkan dan pesan hanya dikonsumsi sesuai permintaan.

Parameter	Nilai default	Deskripsi
`maxInflightOutboundBatches`	5 batch	Jumlah maksimum batch keluar aktif yang dapat diproses secara bersamaan. Semakin tinggi pengaturan, batch keluar yang lebih cepat dapat dikirim (tunduk pada kuota seperti CPU atau bandwidth) dan semakin banyak thread yang dikonsumsi oleh. `AmazonSQSBufferedAsyncClient`

Parameter	Nilai default	Deskripsi
`maxInflightReceiveBatches`	10 batch	Jumlah maksimum batch penerima aktif yang dapat diproses secara bersamaan. Semakin tinggi pengaturan, semakin banyak pesan yang dapat diterima (tunduk pada kuota seperti CPU atau bandwidth), dan semakin banyak utas yang dikonsumsi oleh `AmazonSQSBufferedAsyncClient` . **Note** `0` menunjukkan bahwa semua pra-pengambilan pesan dinonaktifkan dan pesan hanya dikonsumsi sesuai permintaan.

Parameter	Nilai default	Deskripsi
`visibilityTimeoutSeconds`	-1	Ketika parameter ini disetel ke nilai positif, bukan nol, batas waktu visibilitas yang ditetapkan di sini akan mengganti batas waktu visibilitas yang ditetapkan pada antrian dari mana pesan dikonsumsi. **Note** `-1`menunjukkan bahwa pengaturan default dipilih untuk antrian. Anda tidak dapat mengatur batas waktu visibilitas ke. `0`

Meningkatkan throughput menggunakan penskalaan horizontal dan batching aksi dengan Amazon SQS

Antrian Amazon SQS dapat memberikan throughput yang sangat tinggi. Untuk informasi tentang kuota throughput, lihat. Kuota pesan Amazon SQS

Untuk mencapai throughput yang tinggi, Anda harus mengukur produsen dan konsumen pesan secara horizontal (tambahkan lebih banyak produsen dan konsumen).

Topik

- Penskalaan horizontal
- Aksi batching
- Contoh Java yang berfungsi untuk permintaan operasi tunggal dan batch

Penskalaan horizontal

Karena Anda mengakses Amazon SQS melalui protokol permintaan-respons HTTP, latensi permintaan (interval antara memulai permintaan dan menerima respons) membatasi throughput yang dapat Anda capai dari satu utas menggunakan satu koneksi. Misalnya, jika latensi dari klien berbasis Amazon EC2 ke Amazon SQS di wilayah yang sama rata-rata 20 ms, throughput maksimum dari satu utas melalui satu koneksi rata-rata 50 TPS.

Penskalaan horizontal melibatkan peningkatan jumlah produsen pesan (yang membuat `SendMessage` permintaan) dan konsumen (yang membuat `ReceiveMessage` dan `DeleteMessage` meminta) untuk meningkatkan throughput antrian Anda secara keseluruhan. Anda dapat menskalakan secara horizontal dengan tiga cara:

- Meningkatkan jumlah thread per klien
- Tambahkan lebih banyak klien
- Tingkatkan jumlah thread per klien dan tambahkan lebih banyak klien

Ketika Anda menambahkan lebih banyak klien, Anda mencapai keuntungan linier pada dasarnya dalam throughput antrian. Misalnya, jika Anda menggandakan jumlah klien, Anda juga menggandakan throughput.

> **Note**
>
> Saat Anda menskalakan secara horizontal, pastikan klien Amazon SQS Anda memiliki koneksi atau utas yang cukup untuk mendukung jumlah produsen dan konsumen pesan bersamaan yang mengirim permintaan dan menerima tanggapan. Misalnya, secara default, instance AWS SDK for Java `AmazonSQSClient` kelas mempertahankan paling banyak 50 koneksi ke Amazon SQS. Untuk membuat produsen dan konsumen bersamaan tambahan, Anda harus menyesuaikan jumlah maksimum benang produsen dan konsumen yang diijinkan pada suatu `AmazonSQSClientBuilder` objek, misalnya:
>
> ```
> final AmazonSQS sqsClient = AmazonSQSClientBuilder.standard()
>         .withClientConfiguration(new ClientConfiguration()
>                 .withMaxConnections(producerCount + consumerCount))
>         .build();
> ```
>
> Untuk`AmazonSQSAsyncClient`, Anda juga harus memastikan bahwa cukup banyak utas yang tersedia.

Contoh ini hanya berfungsi untuk Java v. 1.x.

Aksi batching

Batching melakukan lebih banyak pekerjaan selama setiap perjalanan pulang pergi ke layanan (misalnya, ketika Anda mengirim beberapa pesan dengan satu `SendMessageBatch` permintaan). Tindakan batch Amazon SQS adalah`SendMessageBatch`,`DeleteMessageBatch`, dan. `ChangeMessageVisibilityBatch` Untuk memanfaatkan batching tanpa mengubah produsen atau konsumen, Anda dapat menggunakan Amazon SQS Buffered Asynchronous Client.

> **Note**
>
> Karena `ReceiveMessage` dapat memproses 10 pesan sekaligus, tidak ada `ReceiveMessageBatch` tindakan.

Batching mendistribusikan latensi tindakan batch melalui beberapa pesan dalam permintaan batch, daripada menerima seluruh latensi untuk satu pesan (misalnya, permintaan). `SendMessage` Karena setiap perjalanan pulang pergi membawa lebih banyak pekerjaan, permintaan batch membuat penggunaan thread dan koneksi lebih efisien, meningkatkan throughput.

Anda dapat menggabungkan batching dengan penskalaan horizontal untuk menyediakan throughput dengan thread, koneksi, dan permintaan yang lebih sedikit daripada permintaan pesan individual. Anda dapat menggunakan tindakan Amazon SQS batch untuk mengirim, menerima, atau menghapus hingga 10 pesan sekaligus. Karena Amazon SQS mengenakan biaya berdasarkan permintaan, batching dapat secara substansional mengurangi biaya Anda.

Batching dapat menimbulkan beberapa kerumitan untuk aplikasi Anda (misalnya, aplikasi Anda harus mengumpulkan pesan sebelum mengirimnya, atau terkadang harus menunggu lebih lama untuk respons). Namun, batching masih bisa efektif dalam kasus-kasus berikut:

- Aplikasi Anda menghasilkan banyak pesan dalam waktu singkat, sehingga penundaan tidak pernah terlalu lama.
- Konsumen pesan mengambil pesan dari antrian atas kebijakannya sendiri, tidak seperti produsen pesan biasa yang perlu mengirim pesan sebagai respons terhadap peristiwa yang tidak mereka kendalikan.

> **Important**
>
> Permintaan batch mungkin berhasil meskipun pesan individual dalam batch gagal. Setelah permintaan batch, selalu periksa kegagalan pesan individual dan coba lagi tindakan jika perlu.

Contoh Java yang berfungsi untuk permintaan operasi tunggal dan batch

Prasyarat

Tambahkan`aws-java-sdk-sqs.jar`,`aws-java-sdk-ec2.jar`, dan `commons-logging.jar` paket ke jalur kelas build Java Anda. Contoh berikut menampilkan dependensi ini dalam file `pom.xml` proyek 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

Contoh kode Java berikut mengimplementasikan pola produsen-konsumen sederhana. Thread utama memunculkan sejumlah thread produsen dan konsumen yang memproses pesan 1 KB untuk waktu tertentu. Contoh ini mencakup produsen dan konsumen yang membuat permintaan operasi tunggal dan mereka yang membuat permintaan batch.

```
/*
```

```
 * Copyright 2010-2024 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.
            }
        }
    }
}
```

Memantau metrik volume dari contoh yang dijalankan

Amazon SQS secara otomatis menghasilkan metrik volume untuk pesan yang dikirim, diterima, dan dihapus. Anda dapat mengakses metrik tersebut dan lainnya melalui tab Monitoring untuk antrian Anda atau di CloudWatch konsol.

> **Note**
>
> Metrik dapat memakan waktu hingga 15 menit setelah antrian mulai tersedia.

Bekerja dengan JMS dan Amazon SQS

Amazon SQS Java Messaging Library adalah antarmuka Java Message Service (JMS) untuk Amazon SQS yang memungkinkan Anda memanfaatkan Amazon SQS dalam aplikasi yang sudah menggunakan JMS. Antarmuka memungkinkan Anda menggunakan Amazon SQS sebagai penyedia JMS dengan perubahan kode minimal. Bersama dengan AWS SDK for Java, Amazon SQS Java Messaging Library memungkinkan Anda membuat koneksi dan sesi JMS, serta produsen dan konsumen yang mengirim dan menerima pesan ke dan dari antrian Amazon SQS.

Pustaka mendukung pengiriman dan penerimaan pesan ke antrian (point-to-pointmodel JMS) sesuai dengan spesifikasi JMS 1.1. Pustaka mendukung pengiriman pesan teks, byte, atau objek secara sinkron ke antrian Amazon SQS. Pustaka juga mendukung penerimaan objek secara sinkron atau asinkron.

Untuk informasi tentang fitur Amazon SQS Java Messaging Library yang mendukung spesifikasi JMS 1.1, lihat Amazon SQS mendukung implementasi JMS 1.1 dan FAQ Amazon SQS.

Topik

Prasyarat untuk bekerja dengan JMS dan Amazon SQS

Sebelum Anda mulai, Anda harus memiliki prasyarat berikut:

- SDK for Java

  Ada dua cara untuk menyertakan SDK for Java dalam proyek Anda:

  - Unduh dan instal SDK for Java.
  - Gunakan Maven untuk mendapatkan Amazon SQS Java Messaging Library.

> **Note**
>
> SDK for Java disertakan sebagai dependensi.
> SDK for Java dan Amazon SQS Extended Client Library untuk Java memerlukan J2SE Development Kit 8.0 atau yang lebih baru.

Untuk informasi tentang mengunduh SDK for Java, lihat SDK for Java.

- Perpustakaan Pesan Amazon SQS Java

  Jika Anda tidak menggunakan Maven, Anda harus menambahkan `amazon-sqs-java-messaging-lib.jar` paket ke jalur kelas Java. Untuk informasi tentang mengunduh pustaka, lihat Amazon SQS Java Messaging Library.

  > **Note**
  >
  > Amazon SQS Java Messaging Library mencakup dukungan untuk Maven dan Spring Framework.
  > Untuk contoh kode yang menggunakan Maven, Spring Framework, dan Amazon SQS Java Messaging Library, lihat. Contoh Java yang berfungsi untuk menggunakan JMS dengan antrian standar Amazon SQS
  >
  > ```
  > <dependency>
  >   <groupId>com.amazonaws</groupId>
  >   <artifactId>amazon-sqs-java-messaging-lib</artifactId>
  >   <version>1.0.4</version>
  >   <type>jar</type>
  > </dependency>
  > ```

- Antrian Amazon SQS

  Buat antrian menggunakan AWS Management Console for Amazon SQS, API, atau klien Amazon SQS `CreateQueue` yang dibungkus yang disertakan dalam Amazon SQS Java Messaging Library.

  - Untuk informasi tentang membuat antrian dengan Amazon SQS menggunakan API atau `CreateQueue` API, lihat Membuat Antrian. AWS Management Console

- Untuk informasi tentang menggunakan Amazon SQS Java Messaging Library, lihat. Memulai dengan Amazon SQS Java Messaging Library

Memulai dengan Amazon SQS Java Messaging Library

Untuk mulai menggunakan Java Message Service (JMS) dengan Amazon SQS, gunakan contoh kode di bagian ini. Bagian berikut menunjukkan cara membuat koneksi JMS dan sesi, dan cara mengirim dan menerima pesan.

Objek klien Amazon SQS yang dibungkus yang disertakan dalam Perpustakaan Pesan Java Amazon SQS memeriksa apakah antrian Amazon SQS ada. Jika antrian tidak ada, klien membuatnya.

Membuat koneksi JMS

1. Buat pabrik koneksi dan panggil `createConnection` metode melawan pabrik.

```
// 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();
```

   `SQSConnection`Kelas meluas. `javax.jms.Connection` Bersama dengan metode koneksi standar JMS, `SQSConnection` menawarkan metode tambahan, seperti `getAmazonSQSClient` dan`getWrappedAmazonSQSClient`. Kedua metode memungkinkan Anda melakukan operasi administratif yang tidak termasuk dalam spesifikasi JMS, seperti membuat antrian baru. Namun, `getWrappedAmazonSQSClient` metode ini juga menyediakan versi terbungkus dari klien Amazon SQS yang digunakan oleh koneksi saat ini. Pembungkus mengubah setiap pengecualian dari klien menjadi`JMSException`, memungkinkannya untuk lebih mudah digunakan oleh kode yang ada yang mengharapkan `JMSException` kejadian.

2. Anda dapat menggunakan objek klien yang dikembalikan dari `getAmazonSQSClient` dan `getWrappedAmazonSQSClient` untuk melakukan operasi administratif yang tidak termasuk dalam spesifikasi JMS (misalnya, Anda dapat membuat antrean Amazon SQS).

Jika Anda memiliki kode yang mengharapkan pengecualian JMS, maka Anda harus menggunakan: `getWrappedAmazonSQSClient`

- Jika Anda menggunakan`getWrappedAmazonSQSClient`, objek klien yang dikembalikan mengubah semua pengecualian menjadi pengecualian JMS.
- Jika Anda menggunakan`getAmazonSQSClient`, pengecualian adalah semua pengecualian Amazon SQS.

Membuat antrian Amazon SQS

Objek klien yang dibungkus memeriksa apakah antrian Amazon SQS ada.

Jika antrian tidak ada, klien membuatnya. Jika antrian memang ada, fungsi tidak mengembalikan apa pun. Untuk informasi selengkapnya, lihat bagian “Buat antrian jika diperlukan” pada TextMessageSender.java contoh.

Untuk membuat antrian standar

```
// 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");
}
```

Untuk membuat antrian 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**
>
> Nama antrian FIFO harus diakhiri dengan akhiran. `.fifo`
> Untuk informasi selengkapnya tentang `ContentBasedDeduplication` atribut, lihatTepat sekali diproses di Amazon SQS.

Mengirim pesan secara sinkron

1. Saat koneksi dan antrean Amazon SQS yang mendasarinya sudah siap, buat sesi JMS yang tidak ditransaksikan dengan mode. `AUTO_ACKNOWLEDGE`

```
// Create the nontransacted session with AUTO_ACKNOWLEDGE mode
Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
```

2. Untuk mengirim pesan teks ke antrian, buat identitas antrian JMS dan produser pesan.

```
// 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. Buat pesan teks dan kirimkan ke antrian.

   - Untuk mengirim pesan ke antrian standar, Anda tidak perlu mengatur parameter tambahan apa pun.

```
// Create the text message
TextMessage message = session.createTextMessage("Hello World!");

// Send the message
producer.send(message);
System.out.println("JMS Message " + message.getJMSMessageID());
```

   - Untuk mengirim pesan ke antrian FIFO, Anda harus mengatur ID grup pesan. Anda juga dapat mengatur ID deduplikasi pesan. Untuk informasi selengkapnya, lihat Istilah kunci Amazon SQS.

```
// 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"));
```

Menerima pesan secara sinkron

1. Untuk menerima pesan, buat konsumen untuk antrian yang sama dan panggil metode. `start`

   Anda dapat memanggil `start` metode pada koneksi kapan saja. Namun, konsumen tidak mulai menerima pesan sampai Anda memanggilnya.

   ```
   // Create a consumer for the 'MyQueue'
   MessageConsumer consumer = session.createConsumer(queue);
   // Start receiving incoming messages
   connection.start();
   ```

2. Panggil `receive` metode pada konsumen dengan batas waktu diatur ke 1 detik, dan kemudian cetak konten pesan yang diterima.

   - Setelah menerima pesan dari antrian standar, Anda dapat mengakses konten pesan.

     ```
     // 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());
     }
     ```

- Setelah menerima pesan dari antrian FIFO, Anda dapat mengakses konten pesan dan atribut pesan khusus FIFO lainnya, seperti ID grup pesan, ID deduplikasi pesan, dan nomor urut. Untuk informasi selengkapnya, lihat Istilah kunci Amazon SQS.

```
// 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. Tutup koneksi dan sesi.

```
// Close the connection (and the session).
connection.close();
```

Output akan terlihat serupa dengan yang berikut ini:

```
JMS Message ID:8example-588b-44e5-bbcf-d816example2
Received: Hello World!
```

> **Note**
>
> Anda dapat menggunakan Spring Framework untuk menginisialisasi objek-objek ini. Untuk informasi tambahan, lihat `SpringExampleConfiguration.xmlSpringExample.java`,, dan kelas pembantu lainnya di `ExampleConfiguration.java` dan `ExampleCommon.java` di Contoh Java yang berfungsi untuk menggunakan JMS dengan antrian standar Amazon SQS bagian.

Untuk contoh lengkap mengirim dan menerima objek, lihat TextMessageSender.java danSyncMessageReceiver.java.

Menerima pesan secara asinkron

Dalam contoh diMemulai dengan Amazon SQS Java Messaging Library, pesan dikirim ke `MyQueue` dan diterima secara serempak.

Contoh berikut menunjukkan cara menerima pesan secara asinkron melalui pendengar.

1. Menerapkan `MessageListener` antarmuka.

```
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();
        }
    }
}
```

   onMessageMetode `MessageListener` antarmuka dipanggil ketika Anda menerima pesan. Dalam implementasi listener ini, teks yang disimpan dalam pesan dicetak.

2. Alih-alih secara eksplisit memanggil `receive` metode pada konsumen, atur pendengar pesan konsumen ke instance implementasi. `MyListener` Utas utama menunggu satu detik.

```
// 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);
```

Langkah-langkah lainnya identik dengan yang ada di Memulai dengan Amazon SQS Java Messaging Library contoh. Untuk contoh lengkap konsumen asinkron, lihat di. `AsyncMessageReceiver.java` Contoh Java yang berfungsi untuk menggunakan JMS dengan antrian standar Amazon SQS

Output untuk contoh ini terlihat mirip dengan yang berikut:

```
JMS Message ID:8example-588b-44e5-bbcf-d816example2
Received: Hello World!
```

Menggunakan mode pengakuan klien

Contoh dalam `AUTO_ACKNOWLEDGE` mode Memulai dengan Amazon SQS Java Messaging Library penggunaan di mana setiap pesan yang diterima diakui secara otomatis (dan karenanya dihapus dari antrian Amazon SQS yang mendasarinya).

1. Untuk secara eksplisit mengakui pesan setelah diproses, Anda harus membuat sesi dengan mode. `CLIENT_ACKNOWLEDGE`

   ```
   // Create the non-transacted session with CLIENT_ACKNOWLEDGE mode.
   Session session = connection.createSession(false, Session.CLIENT_ACKNOWLEDGE);
   ```

2. Ketika pesan diterima, tampilkan dan kemudian secara eksplisit mengakuinya.

   ```
   // 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**
   >
   > Dalam mode ini, ketika pesan diakui, semua pesan yang diterima sebelum pesan ini secara implisit diakui juga. Misalnya, jika 10 pesan diterima, dan hanya pesan ke-10 yang diakui (dalam urutan pesan diterima), maka semua dari sembilan pesan sebelumnya juga diakui.

Langkah-langkah lainnya identik dengan yang ada di Memulai dengan Amazon SQS Java Messaging Library contoh. Untuk contoh lengkap konsumen sinkron dengan mode pengakuan klien, lihat `SyncMessageReceiverClientAcknowledge.java` di. Contoh Java yang berfungsi untuk menggunakan JMS dengan antrian standar Amazon SQS

Output untuk contoh ini terlihat mirip dengan yang berikut:

```
JMS Message ID:4example-aa0e-403f-b6df-5e02example5
Received: Hello World!
Acknowledged: ID:4example-aa0e-403f-b6df-5e02example5
```

Menggunakan mode pengakuan tidak berurutan

Saat menggunakan `CLIENT_ACKNOWLEDGE` mode, semua pesan yang diterima sebelum pesan yang diakui secara eksplisit diakui secara otomatis. Untuk informasi selengkapnya, lihat Menggunakan mode pengakuan klien.

Amazon SQS Java Messaging Library menyediakan mode pengakuan lain. Saat menggunakan `UNORDERED_ACKNOWLEDGE` mode, semua pesan yang diterima harus diakui secara individual dan eksplisit oleh klien, terlepas dari pesanan penerimaan mereka. Untuk melakukan ini, buat sesi dengan `UNORDERED_ACKNOWLEDGE` mode.

```
// Create the non-transacted session with UNORDERED_ACKNOWLEDGE mode.
Session session = connection.createSession(false, SQSSession.UNORDERED_ACKNOWLEDGE);
```

Langkah-langkah yang tersisa identik dengan yang ada di Menggunakan mode pengakuan klien contoh. Untuk contoh lengkap konsumen sinkron dengan `UNORDERED_ACKNOWLEDGE` mode, lihat`SyncMessageReceiverUnorderedAcknowledge.java`.

Dalam contoh ini, outputnya terlihat mirip dengan yang berikut:

```
JMS Message ID:dexample-73ad-4adb-bc6c-4357example7
Received: Hello World!
Acknowledged: ID:dexample-73ad-4adb-bc6c-4357example7
```

Menggunakan Java Message Service dengan klien Amazon SQS lainnya

Menggunakan Amazon SQS Java Message Service (JMS) Client dengan AWS SDK membatasi ukuran pesan Amazon SQS hingga 256 KB. Namun, Anda dapat membuat penyedia JMS menggunakan klien Amazon SQS apa pun. Misalnya, Anda dapat menggunakan Klien JMS dengan Amazon SQS Extended Client Library for Java untuk mengirim pesan Amazon SQS yang berisi referensi ke payload pesan (hingga 2 GB) di Amazon S3. Untuk informasi selengkapnya, lihat Mengelola pesan Amazon SQS besar menggunakan Java dan Amazon S3.

Contoh kode Java berikut membuat penyedia JMS untuk Extended Client Library:

```
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);
```

Contoh kode Java berikut membuat pabrik koneksi:

```
// 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();
```

Contoh Java yang berfungsi untuk menggunakan JMS dengan antrian standar Amazon SQS

Contoh kode berikut menunjukkan cara menggunakan Java Message Service (JMS) dengan antrian standar Amazon SQS. Untuk informasi lebih lanjut tentang bekerja dengan antrian FIFO, lihatUntuk membuat antrian FIFO,, dan. Mengirim pesan secara sinkron Menerima pesan secara sinkron (Menerima pesan secara sinkron sama untuk antrian standar dan FIFO. Namun, pesan dalam antrian FIFO berisi lebih banyak atribut.)

ExampleConfiguration.jawa

Contoh kode Java SDK v 1.x berikut menetapkan nama antrian default, wilayah, dan kredenal yang akan digunakan dengan contoh Java lainnya.

```
/*
 * Copyright 2010-2024 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

Contoh kode Java berikut menciptakan produser pesan teks.

```
/*
 * Copyright 2010-2024 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

Contoh kode Java berikut menciptakan konsumen pesan sinkron.

```
/*
 * Copyright 2010-2024 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

Contoh kode Java berikut menciptakan konsumen pesan asinkron.

```
/*
 * Copyright 2010-2024 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

Contoh kode Java berikut menciptakan konsumen sinkron dengan modus mengakui klien.

```
/*
 * Copyright 2010-2024 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

Contoh kode Java berikut menciptakan konsumen sinkron dengan modus mengakui tidak berurutan.

```
/*
 * Copyright 2010-2024 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

Contoh kode XHTML berikut adalah file konfigurasi kacang untukSpringExample.jawa.

```
<!--
    Copyright 2010-2024 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.jawa

Contoh kode Java berikut menggunakan file konfigurasi bean untuk menginisialisasi objek Anda.

```
/*
 * Copyright 2010-2024 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.jawa

Contoh kode Java berikut memeriksa apakah antrian Amazon SQS ada dan kemudian membuatnya jika tidak. Ini juga termasuk contoh kode logging.

```
/*
 * Copyright 2010-2024 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() );
        }
    }
}
```

Amazon SQS mendukung implementasi JMS 1.1

Amazon SQS Java Messaging Library mendukung implementasi JMS 1.1 berikut. Untuk informasi selengkapnya tentang fitur dan kemampuan yang didukung dari Amazon SQS Java Messaging Library, lihat FAQ Amazon SQS.

Antarmuka umum yang didukung

- `Connection`
- `ConnectionFactory`
- `Destination`
- `Session`
- `MessageConsumer`
- `MessageProducer`

Jenis pesan yang didukung

- `ByteMessage`
- `ObjectMessage`

- `TextMessage`

Mode pengakuan pesan yang didukung

- `AUTO_ACKNOWLEDGE`
- `CLIENT_ACKNOWLEDGE`
- `DUPS_OK_ACKNOWLEDGE`
- `UNORDERED_ACKNOWLEDGE`

> **Note**
>
> `UNORDERED_ACKNOWLEDGE`Mode ini bukan bagian dari spesifikasi JMS 1.1. Mode ini membantu Amazon SQS mengizinkan klien JMS untuk secara eksplisit mengakui pesan.

Header yang ditentukan JMS dan properti yang dicadangkan

Untuk mengirim pesan

Saat mengirim pesan, Anda dapat mengatur header dan properti berikut untuk setiap pesan:

- `JMSXGroupID`(diperlukan untuk antrian FIFO, tidak diperbolehkan untuk antrian standar)
- `JMS_SQS_DeduplicationId`(opsional untuk antrian FIFO, tidak diperbolehkan untuk antrian standar)

Setelah Anda mengirim pesan, Amazon SQS menetapkan header dan properti berikut untuk setiap pesan:

- `JMSMessageID`
- `JMS_SQS_SequenceNumber`(hanya untuk antrian FIFO)

Untuk menerima pesan

Saat Anda menerima pesan, Amazon SQS menetapkan header dan properti berikut untuk setiap pesan:

- `JMSDestination`
- `JMSMessageID`
- `JMSRedelivered`
- `JMSXDeliveryCount`
- `JMSXGroupID`(hanya untuk antrian FIFO)
- `JMS_SQS_DeduplicationId`(hanya untuk antrian FIFO)
- `JMS_SQS_SequenceNumber`(hanya untuk antrian FIFO)

Tutorial Amazon SQS

Bagian ini menyediakan tutorial yang dapat Anda gunakan untuk menjelajahi fitur dan fungsionalitas Amazon SQS.

Topik

Membuat antrian Amazon SQS menggunakan AWS CloudFormation

Anda dapat menggunakan AWS CloudFormation konsol dan template JSON (atau YAMAL) untuk membuat antrean Amazon SQS. Untuk informasi selengkapnya, lihat Bekerja dengan AWS CloudFormation Template dan `AWS::SQS::Queue`Sumber Daya di Panduan AWS CloudFormation Pengguna.

Untuk digunakan AWS CloudFormation untuk membuat antrean Amazon SQS.

1. Salin kode JSON berikut ke file bernama `MyQueue.json`. Untuk membuat antrian standar, hilangkan properti `FifoQueue` dan`ContentBasedDeduplication`. Untuk informasi lebih lanjut tentang deduplikasi berbasis konten, lihat. Tepat sekali diproses di Amazon SQS

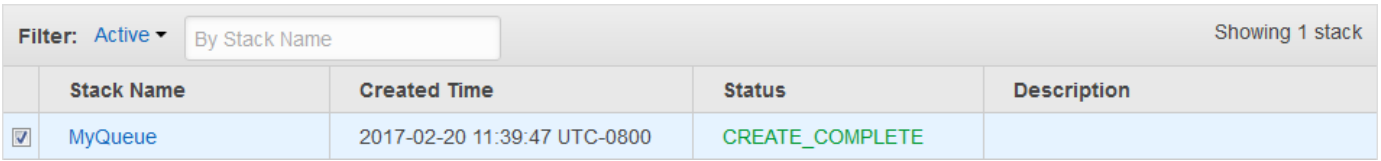
   > **Note**
   >
   > Nama antrian FIFO harus diakhiri dengan akhiran. `.fifo`

```
{
   "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. Masuk ke AWS CloudFormation konsol, lalu pilih Create Stack.
3. Pada panel Tentukan Templat, pilih Unggah file templat, pilih `MyQueue.json` file Anda, lalu pilih Berikutnya.
4. Pada halaman Tentukan Detail, `MyQueue` ketik Nama Tumpukan, lalu pilih Berikutnya.
5. Pada halaman Opsi, pilih Selanjutnya.
6. Pada halaman Tinjau, pilih Buat.

   AWS CloudFormation mulai membuat `MyQueue` tumpukan dan menampilkan status CREATE_IN_PROGRESS. Saat proses selesai, AWS CloudFormation menampilkan status CREATE_COMPLETE.



7. (Opsional) Untuk menampilkan nama, URL, dan ARN antrian, pilih nama tumpukan dan kemudian pada halaman berikutnya perluas bagian Output.

Tutorial: Mengirim pesan ke antrian Amazon SQS dari Amazon Virtual Private Cloud

Dalam tutorial ini, Anda mempelajari cara mengirim pesan ke antrian Amazon SQS melalui jaringan pribadi yang aman. Jaringan ini terdiri dari VPC yang berisi instans Amazon EC2. Instans terhubung ke Amazon SQS melalui titik akhir VPC antarmuka, memungkinkan Anda untuk terhubung ke instans Amazon EC2 dan mengirim pesan ke antrian Amazon SQS meskipun jaringan terputus dari internet publik. Untuk informasi selengkapnya, lihat Titik akhir Amazon Virtual Private Cloud untuk Amazon SQS.

Important

- Anda dapat menggunakan Amazon Virtual Private Cloud hanya dengan titik akhir HTTPS Amazon SQS.
- Saat mengonfigurasi Amazon SQS untuk mengirim pesan dari Amazon VPC, Anda harus mengaktifkan DNS pribadi dan menentukan titik akhir dalam format. `sqs.us-east-2.amazonaws.com`
- DNS pribadi tidak mendukung titik akhir lama seperti atau. `queue.amazonaws.com` `us-east-2.queue.amazonaws.com`

Topik

Langkah 1: Buat pasangan kunci Amazon EC2

Sebuah key pair memungkinkan Anda terhubung ke instans Amazon EC2. Ini terdiri dari kunci publik yang mengenkripsi informasi login Anda dan kunci pribadi yang mendekripsi itu.

1. Masuk ke konsol Amazon EC2.
2. Pada menu navigasi, di bawah Jaringan & Keamanan, pilih Pasangan Kunci.
3. Pilih Buat pasangan kunci.
4. Dalam Buat Pasangan Kunci kotak dialog, untuk Nama pasangan kunci, masukkan`SQS-VPCE-Tutorial-Key-Pair`, lalu pilih Buat.
5. Browser Anda mengunduh file kunci pribadi `SQS-VPCE-Tutorial-Key-Pair.pem` secara otomatis.

   > **Important**
   >
   > Simpan file ini di tempat yang aman. EC2 tidak menghasilkan `.pem` file untuk key pair yang sama untuk kedua kalinya.

6. Untuk memungkinkan klien SSH terhubung ke instans EC2 Anda, atur izin untuk file kunci pribadi Anda sehingga hanya pengguna Anda yang dapat memiliki izin membaca untuk itu, misalnya:

   ```
   chmod 400 SQS-VPCE-Tutorial-Key-Pair.pem
   ```

Langkah 2: Buat AWS sumber daya

Untuk menyiapkan infrastruktur yang diperlukan, Anda harus menggunakan AWS CloudFormation template, yang merupakan cetak biru untuk membuat tumpukan yang terdiri dari AWS sumber daya, seperti instans Amazon EC2 dan antrian Amazon SQS.

Tumpukan untuk tutorial ini mencakup sumber daya berikut:

- VPC dan sumber daya jaringan terkait, termasuk subnet, grup keamanan, gateway internet, dan tabel rute
- Instans Amazon EC2 diluncurkan ke subnet VPC
- Antrean Amazon SQS

1. Unduh AWS CloudFormation template bernama SQS-VPCE-Tutorial-CloudFormation.yamldariGitHub.
2. Masuk ke konsol AWS CloudFormation.
3. Pilih Buat Tumpukan.
4. Pada halaman Pilih Template, pilih Unggah templat ke Amazon S3, pilih **SQS-VPCE-SQS-Tutorial-CloudFormation.yaml** file, lalu pilih Berikutnya.
5. Di halaman Tentukan Detail, lakukan hal berikut:
   a. Untuk Nama tumpukan, masukkan SQS-VPCE-Tutorial-Stack.
   b. Untuk KeyName, pilih SQS-VPCE-Tutorial-Key-Pair.
   c. Pilih Selanjutnya.
6. Pada halaman Opsi, pilih Selanjutnya.
7. Pada halaman Tinjauan, di bagian Kemampuan, pilih Saya mengakui yang AWS CloudFormation mungkin membuat sumber daya IAM dengan nama khusus. , dan kemudian pilih Buat.

AWS CloudFormation mulai membuat tumpukan dan menampilkan status CREATE_IN_PROGRESS. Saat proses selesai, AWS CloudFormation menampilkan status CREATE_COMPLETE.

Langkah 3: Konfirmasikan bahwa instans EC2 Anda tidak dapat diakses publik

AWS CloudFormation Template Anda meluncurkan instans EC2 yang diberi nama SQS-VPCE-Tutorial-EC2-Instance ke dalam VPC Anda. Instans EC2 ini tidak mengizinkan lalu lintas keluar dan tidak dapat mengirim pesan ke Amazon SQS. Untuk memverifikasi ini, Anda harus terhubung ke instance, mencoba menyambung ke titik akhir publik, dan kemudian mencoba mengirim pesan kepada Amazon SQS.

1. Masuk ke konsol Amazon EC2.
2. Pada menu navigasi, di bawah Instans, pilih Instans.
3. Pilih SQS-VPCE -. Tutorial-EC2Instance
4. Salin nama host di bawah DNS Publik (IPv4), misalnya, ec2-203-0-113-0.us-west-2.compute.amazonaws.com.
5. Dari direktori yang berisi key pair yang Anda buat sebelumnya, sambungkan ke instance menggunakan perintah berikut, misalnya:

```
ssh -i SQS-VPCE-Tutorial-Key-Pair.pem ec2-user@ec2-203-0-113-0.us-
east-2.compute.amazonaws.com
```

6. Cobalah untuk terhubung ke titik akhir publik apa pun, misalnya:

   ```
   ping amazon.com
   ```

   Upaya koneksi gagal, seperti yang diharapkan.

7. Masuk ke konsol Amazon SQS.
8. Dari daftar antrian, pilih antrian yang dibuat oleh AWS CloudFormation template Anda, misalnya, vpce-sqs-tutorial-stack-cfqueue-1abcdefgh2ijk.
9. Pada tabel Detail, salin URL, misalnya, https://sqs.us-east-2.amazonaws.com/123456789012/.
10. Dari instans EC2 Anda, coba publikasikan pesan ke antrian menggunakan perintah berikut, misalnya:

    ```
    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."
    ```

    Upaya pengiriman gagal, seperti yang diharapkan.

    > **Important**
    >
    > Kemudian, saat Anda membuat titik akhir VPC untuk Amazon SQS, upaya pengiriman Anda akan berhasil.

Langkah 4: Buat titik akhir Amazon VPC untuk Amazon SQS

Untuk menghubungkan VPC Anda ke Amazon SQS, Anda harus menentukan titik akhir VPC antarmuka. Setelah menambahkan titik akhir, Anda dapat menggunakan Amazon SQS API dari instans EC2 di VPC Anda. Ini memungkinkan Anda mengirim pesan ke antrian dalam AWS jaringan tanpa melintasi internet publik.

> **Note**
>
> Instans EC2 masih belum memiliki akses ke AWS layanan dan titik akhir lain di internet.

1. Masuk ke Amazon konsol Amazon VPC.
2. Pada menu navigasi, pilih Endpoints.
3. Pilih Buat Titik Akhir.
4. Pada halaman Buat Titik Akhir, untuk Nama Layanan, pilih nama layanan untuk Amazon SQS.

   > **Note**
   >
   > Nama layanan bervariasi berdasarkan AWS Wilayah saat ini. Misalnya, jika Anda berada di AS Timur (Ohio), nama layanannya adalah com.amazonaws. ***us-east-2*** .sqs.

5. Untuk VPC, pilih SQS-VPCE-Tutorial-VPC.
6. Untuk Subnet, pilih subnet yang Subnetnya ID berisi SQS-VPCE-Tutorial-Subnet.
7. Untuk grup Keamanan, pilih Pilih grup keamanan, lalu pilih grup keamanan yang Nama grupnya berisi SQS VPCE Tutorial Security Group.
8. Pilih Buat Titik Akhir.

   Titik akhir VPC antarmuka dibuat dan ID-nya ditampilkan, misalnya, vpce-0ab1cdef2ghi3j456k.
9. Pilih Tutup.

   Konsol Amazon VPC membuka halaman Titik akhir.

Amazon VPC mulai membuat titik akhir dan menampilkan status tertunda. Ketika proses selesai, Amazon VPC menampilkan status yang tersedia.

Langkah 5: Kirim pesan ke antrian Amazon SQS Anda

Sekarang VPC Anda menyertakan titik akhir untuk Amazon SQS, Anda dapat terhubung ke instans EC2 dan mengirim pesan ke antrian Anda.

1. Sambungkan kembali ke instans EC2 Anda, misalnya:

```
ssh -i SQS-VPCE-Tutorial-Key-Pair.pem ec2-user@ec2-203-0-113-0.us-
east-2.compute.amazonaws.com
```

2. Cobalah untuk mempublikasikan pesan ke antrian lagi menggunakan perintah berikut, misalnya:

```
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."
```

Upaya pengiriman berhasil dan intisari MD5 dari badan pesan dan ID pesan ditampilkan, misalnya:

```
{
 "MD5OfMessageBody": "a1bcd2ef3g45hi678j90klmn12p34qr5",
 "MessageId": "12345a67-8901-2345-bc67-d890123e45fg"
}
```

Untuk informasi tentang menerima dan menghapus pesan dari antrian yang dibuat oleh AWS CloudFormation template Anda (misalnya, vpce-sqs-tutorial-stack-cfqueue-1abcdefgh2ijk), lihat. Menerima dan menghapus pesan di Amazon SQS

Untuk informasi tentang menghapus sumber daya Anda, lihat berikut ini:

- Menghapus Titik Akhir VPC di Panduan Pengguna Amazon VPC
- Menghapus antrian Amazon SQS
- Menghentikan Instans Anda di Panduan Pengguna Amazon EC2
- Menghapus VPC Anda di Panduan Pengguna Amazon VPC
- Menghapus Tumpukan di AWS CloudFormation Konsol di AWS CloudFormation Panduan Pengguna
- Menghapus Pasangan Kunci Anda di Panduan Pengguna Amazon EC2

Memecahkan masalah di Amazon SQS

Topik berikut memberikan saran pemecahan masalah untuk kesalahan umum dan masalah yang mungkin Anda temui saat menggunakan konsol Amazon SQS, Amazon SQS API, atau alat lain dengan Amazon SQS. Jika Anda menemukan masalah yang tidak tercantum di sini, Anda dapat menggunakan tombol Umpan Balik di halaman ini untuk melaporkannya.

Untuk saran pemecahan masalah selengkapnya dan jawaban atas pertanyaan dukungan umum, kunjungi Pusat Pengetahuan AWS.

Topik

Memecahkan masalah kesalahan akses ditolak di Amazon SQS

Topik berikut mencakup penyebab `AccessDenied` atau `AccessDeniedException` kesalahan paling umum pada panggilan API Amazon SQS. Untuk informasi selengkapnya tentang cara memecahkan masalah kesalahan ini, lihat Bagaimana cara memecahkan masalah kesalahan "" atau AccessDenied "AccessDeniedPengecualian” pada panggilan API Amazon SQS? dalam Panduan Pusat AWS Pengetahuan.

Contoh pesan kesalahan:

```
An error occurred (AccessDenied) when calling the SendMessage operation: Access to
        the resource https://sqs.us-east-1.amazonaws.com/ is denied.
```

- atau -

```
An error occurred (KMS.AccessDeniedException) when calling the SendMessage
```

```
        operation: User: arn:aws:iam::xxxxx:user/xxxx is not authorized to perform:
        kms:GenerateDataKey on resource: arn:aws:kms:us-east-1:xxxx:key/xxxx with an
explicit
        deny.
```

Topik

- Kebijakan antrian Amazon SQS dan kebijakan IAM
- AWS Key Management Service izin
- Kebijakan titik akhir VPC
- Kebijakan kontrol layanan organisasi

Kebijakan antrian Amazon SQS dan kebijakan IAM

Untuk memverifikasi apakah pemohon memiliki izin yang tepat untuk melakukan operasi Amazon SQS, lakukan hal berikut:

- Identifikasi prinsip IAM yang membuat panggilan Amazon SQS API. Jika prinsipal IAM berasal dari akun yang sama, kebijakan antrian Amazon SQS atau AWS kebijakan Identity and Access Management (IAM) and Access Management (IAM) harus menyertakan izin untuk secara eksplisit mengizinkan akses untuk tindakan tersebut.
- Jika prinsipal adalah entitas IAM:
  - Anda dapat mengidentifikasi pengguna atau peran IAM Anda dengan memeriksa sudut kanan atas AWS Management Console, atau dengan menggunakan perintah. `aws sts get-caller-identity`
  - Periksa kebijakan IAM yang terkait dengan peran atau pengguna IAM. Gunakan salah satu metode berikut:
    - Uji kebijakan IAM dengan IAM Policy Simulator.
    - Tinjau berbagai jenis kebijakan IAM yang berbeda.
  - Jika perlu, edit kebijakan pengguna IAM Anda.
  - Periksa kebijakan antrian dan edit jika diperlukan.
- Jika prinsipal adalah AWS layanan, maka kebijakan antrian Amazon SQS harus secara eksplisit mengizinkan akses.
- Jika prinsipal adalah prinsipal lintas akun, maka kebijakan antrian Amazon SQS dan kebijakan IAM harus secara eksplisit mengizinkan akses.

- Jika kebijakan menggunakan elemen kondisi, periksa apakah kondisi membatasi akses.

> **Important**
>
> Penolakan eksplisit di salah satu kebijakan mengesampingkan izin eksplisit. Berikut adalah beberapa contoh dasar kebijakan Amazon SQS.

AWS Key Management Service izin

Jika antrian Amazon SQS Anda mengaktifkan enkripsi sisi server (SSE) dengan pelanggan yang dikelola AWS KMS key, maka izin harus diberikan kepada produsen dan konsumen. Untuk mengonfirmasi apakah antrian dienkripsi, Anda dapat menggunakan **`KmsMasterKeyId`** atribut **`GetQueueAttributes`**API, atau dari konsol antrian di bawah Enkripsi.

- Izin yang diperlukan untuk produsen:

```
{
"Effect": "Allow",
"Action": [
    "kms:GenerateDataKey",
    "kms:Decrypt"
],
"Resource": "<Key ARN>"
}
```

- Izin yang diperlukan untuk konsumen:

```
{
"Effect": "Allow",
"Action": [
    "kms:Decrypt"
],
"Resource": "<Key ARN>"
}
```

- Izin yang diperlukan untuk akses lintas akun:

```
{
"Effect": "Allow",
```

```
"Action": [
    "kms:DescribeKey",
    "kms:Decrypt",
    "kms:ReEncrypt",
    "kms:GenerateDataKey"
],
"Resource": "<Key ARN>"
}
```

Anda dapat menggunakan salah satu dari berikut ini untuk mengaktifkan enkripsi antrian Amazon SQS:

- SSE-Amazon SQS (Kunci enkripsi dibuat dan dikelola oleh layanan Amazon SQS.)
- AWS kunci default terkelola (alias/aws/sqs)
- Kunci yang dikelola pelanggan

Namun, jika Anda menggunakan kunci KMS AWS -managed, Anda tidak dapat mengubah kebijakan kunci default. Oleh karena itu, untuk menyediakan akses ke layanan lain dan lintas akun, gunakan kunci yang dikelola pelanggan. Melakukan hal ini memungkinkan Anda untuk mengedit kebijakan utama.

Kebijakan titik akhir VPC

Jika Anda mengakses Amazon SQS melalui titik akhir Amazon Virtual Private Cloud (Amazon VPC), kebijakan titik akhir Amazon SQS VPC harus mengizinkan akses. Anda dapat membuat kebijakan untuk titik akhir VPC Amazon untuk Amazon SQS, di mana Anda dapat menentukan hal berikut:

1. Prinsipal yang dapat melakukan tindakan.
2. Tindakan yang dapat dilakukan.
3. Sumber daya yang menjadi target tindakan.

Dalam contoh berikut, kebijakan titik akhir VPC menentukan bahwa pengguna IAM diizinkan mengirim pesan ke *MyUser*antrean Amazon SQS. *MyQueue* Tindakan lain, pengguna IAM, dan sumber daya Amazon SQS ditolak aksesnya melalui titik akhir VPC.

```
{
   "Statement": [{
```

```
      "Action": ["sqs:SendMessage"],
      "Effect": "Allow",
      "Resource": "arn:aws:sqs:us-east-2:123456789012:MyQueue",
      "Principal": {
        "AWS": "arn:aws:iam:123456789012:user/MyUser"
      }
   }]
}
```

Kebijakan kontrol layanan organisasi

Jika Anda Akun AWS termasuk dalam organisasi, AWS Organizations kebijakan dapat memblokir Anda dari mengakses antrian Amazon SQS Anda. Secara default, AWS Organizations kebijakan tidak memblokir permintaan apa pun ke Amazon SQS. Namun, pastikan AWS Organizations kebijakan Anda belum dikonfigurasi untuk memblokir akses ke antrian Amazon SQS. Untuk petunjuk tentang cara memeriksa AWS Organizations kebijakan Anda, lihat Mencantumkan semua kebijakan di Panduan AWS Organizations Pengguna.

Memecahkan masalah kesalahan Amazon SQS API

Topik berikut mencakup kesalahan paling umum yang dikembalikan saat melakukan panggilan Amazon SQS API, dan cara memecahkan masalah.

Topik

- QueueDoesNotExist kesalahan
- InvalidAttributeValue kesalahan
- ReceiptHandle kesalahan

QueueDoesNotExist kesalahan

Kesalahan ini akan dikembalikan ketika layanan Amazon SQS tidak dapat menemukan antrian yang disebutkan untuk tindakan Amazon SQS.

Kemungkinan penyebab dan mitigasi:

- Wilayah salah: Tinjau konfigurasi klien Amazon SQS untuk mengonfirmasi bahwa Anda mengonfigurasi Wilayah yang benar pada klien. Bila Anda tidak mengonfigurasi Region pada klien, maka SDK atau AWS CLI memilih Region dari file konfigurasi atau variabel lingkungan. Jika SDK

tidak menemukan Region dalam file konfigurasi, maka SDK menetapkan Region ke us-east-1 secara default.

- Antrian mungkin baru saja dihapus: Jika antrian dihapus sebelum panggilan API dibuat, maka panggilan API akan mengembalikan kesalahan ini. CloudTrailPeriksa DeleteQueueoperasi apa pun sebelum waktu kesalahan.
- Masalah izin: Jika pengguna atau peran yang meminta AWS Identity and Access Management (IAM) tidak memiliki izin yang diperlukan, maka Anda mungkin menerima kesalahan berikut:

```
The specified queue does not exist or you do not have access to it.
```

Periksa izin, dan lakukan panggilan API dengan izin yang benar.

Untuk detail selengkapnya tentang pemecahan masalah `QueueDoesNotExist` kesalahan, lihat Bagaimana cara memecahkan masalah QueueDoesNotExist kesalahan saat melakukan panggilan API ke antrean Amazon SQS saya? dalam Panduan Pusat AWS Pengetahuan.

InvalidAttributeValue kesalahan

Kesalahan ini akan dikembalikan setelah memperbarui kebijakan sumber daya antrian Amazon SQS, atau properti dengan kebijakan atau prinsipal yang salah.

Kemungkinan penyebab dan mitigasi:

- Kebijakan sumber daya tidak valid: Periksa apakah kebijakan sumber daya memiliki semua bidang yang diperlukan. Untuk informasi selengkapnya, lihat referensi elemen kebijakan IAM JSON dan Memvalidasi kebijakan IAM. Anda juga dapat menggunakan generator kebijakan IAM untuk membuat dan menguji kebijakan sumber daya Amazon SQS. Pastikan kebijakan tersebut dalam format JSON.
- Prinsipal tidak valid: Pastikan bahwa `Principal` elemen ada dalam kebijakan sumber daya dan nilainya valid. Jika `Principal` elemen kebijakan sumber daya Amazon SQS menyertakan entitas IAM, pastikan entitas tersebut ada sebelum Anda menggunakan kebijakan tersebut. Amazon SQS memvalidasi kebijakan sumber daya dan memeriksa entitas IAM. Jika entitas IAM tidak ada, Anda akan menerima kesalahan. Untuk mengonfirmasi entitas IAM, gunakan `GetRole`dan `GetUser`API.

Untuk informasi tambahan tentang cara memecahkan masalah `InvalidAttributeValue` kesalahan, lihat Bagaimana cara memecahkan masalah QueueDoesNotExist kesalahan saat melakukan panggilan API ke antrean Amazon SQS saya? dalam Panduan Pusat AWS Pengetahuan.

ReceiptHandle kesalahan

Setelah melakukan panggilan `DeleteMessage`API, kesalahan `ReceiptHandleIsInvalid` atau `InvalidParameterValue` mungkin dikembalikan jika pegangan tanda terima salah atau kedaluwarsa.

- ReceiptHandleIsInvalid error: Jika pegangan tanda terima salah, Anda akan menerima kesalahan yang mirip dengan contoh ini:

```
An error occurred (ReceiptHandleIsInvalid) when calling the DeleteMessage operation:
 The input receipt handle <YOUR RECEIPT HANDLE> is not a valid receipt handle.
```

- InvalidParameterValue error: Jika tanda terima kedaluwarsa, Anda akan menerima kesalahan yang mirip dengan contoh ini:

```
An error occurred (InvalidParameterValue) when calling the DeleteMessage operation:
 Value <YOUR RECEIPT HANDLE> for parameter ReceiptHandle is invalid. Reason: The
 receipt handle has expired.
```

Kemungkinan penyebab dan mitigasi:

Pegangan tanda terima dibuat untuk setiap pesan yang diterima, dan hanya berlaku untuk periode batas waktu visibilitas. Ketika periode batas waktu visibilitas berakhir, pesan menjadi terlihat pada antrian untuk konsumen. Ketika Anda menerima pesan lagi dari konsumen, Anda menerima pegangan tanda terima baru. Untuk mencegah kesalahan penanganan tanda terima yang salah atau kedaluwarsa, gunakan gagang tanda terima yang benar untuk menghapus pesan dalam periode batas waktu visibilitas antrian Amazon SQS.

Untuk informasi tambahan tentang cara memecahkan masalah `ReceiptHandle` kesalahan, lihat Bagaimana cara memecahkan masalah kesalahan "" dan ReceiptHandle IsInvalid "InvalidParameterNilai" saat saya menggunakan panggilan API Amazon SQS? DeleteMessage dalam Panduan Pusat AWS Pengetahuan.

Memecahkan masalah antrian surat mati Amazon SQS dan masalah redrive DLQ

Topik berikut mencakup penyebab paling umum masalah Amazon SQS DLQ dan DLQ redrive, dan cara memecahkan masalah tersebut. Untuk informasi selengkapnya, lihat Bagaimana cara mengatasi masalah redrive Amazon SQS DLQ? dalam Panduan Pusat AWS Pengetahuan.

Topik

- Masalah DLQ
- Masalah DLQ-redrive

Masalah DLQ

Pelajari tentang masalah DLQ umum dan cara mengatasinya.

Topik

- Melihat pesan menggunakan konsol dapat menyebabkan pesan dipindahkan ke antrean huruf mati
- Antrian NumberOfMessagesSent dan NumberOfMessagesReceived untuk surat mati tidak cocok
- Membuat dan mengonfigurasi redrive antrian huruf mati
- Penanganan kegagalan pesan antrian standar dan FIFO

Melihat pesan menggunakan konsol dapat menyebabkan pesan dipindahkan ke antrean huruf mati

Amazon SQS menghitung melihat pesan di konsol terhadap kebijakan redrive antrean terkait. Oleh karena itu, jika Anda melihat pesan di konsol berapa kali ditentukan dalam kebijakan redrive antrian terkait, pesan akan dipindahkan ke antrean huruf mati antrian yang sesuai.

Untuk menyesuaikan perilaku ini, Anda dapat melakukan salah satu hal berikut:

- Meningkatkan pengaturan Penerimaan Maksimum untuk kebijakan redrive antrean terkait.
- Hindari melihat pesan antrian yang sesuai di konsol.

Antrian **NumberOfMessagesSent** dan **NumberOfMessagesReceived** untuk surat mati tidak cocok

Jika Anda mengirim pesan ke antrian huruf mati secara manual, pesan tersebut ditangkap oleh metrik. `NumberOfMessagesSent` Namun, jika pesan dikirim ke antrian huruf mati sebagai akibat dari upaya pemrosesan yang gagal, pesan tersebut tidak ditangkap oleh metrik ini. Oleh karena itu, dimungkinkan untuk nilai-nilai `NumberOfMessagesSent` dan `NumberOfMessagesReceived`untuk menjadi berbeda.

Membuat dan mengonfigurasi redrive antrian huruf mati

Penggerak ulang antrian huruf mati mengharuskan Anda menetapkan izin yang sesuai untuk Amazon SQS untuk menerima pesan dari antrian huruf mati, dan mengirim pesan ke antrean tujuan. Jika Anda tidak memiliki izin yang benar, tugas redrive antrian huruf mati bisa gagal. Anda dapat melihat status tugas penggerak ulang pesan untuk memperbaiki masalah, dan coba lagi.

Penanganan kegagalan pesan antrian standar dan FIFO

Antrian standar terus memproses pesan hingga berakhirnya periode retensi. Pemrosesan berkelanjutan ini meminimalkan kemungkinan antrian diblokir oleh pesan yang tidak dikonsumsi. Memiliki sejumlah besar pesan yang berulang kali gagal dihapus konsumen dapat meningkatkan biaya, dan menempatkan beban ekstra pada perangkat keras. Untuk menekan biaya, pindahkan pesan yang gagal ke antrian surat mati.

Antrian standar juga memungkinkan sejumlah besar pesan dalam penerbangan. Jika sebagian besar pesan Anda tidak dapat dikonsumsi, dan tidak dikirim ke antrian surat mati, kecepatan pemrosesan pesan Anda dapat melambat. Untuk menjaga efisiensi antrian Anda, pastikan aplikasi Anda menangani pemrosesan pesan dengan benar.

Antrian FIFO menyediakan pemrosesan tepat sekali dengan mengkonsumsi pesan secara berurutan dari grup pesan. Oleh karena itu, meskipun konsumen dapat terus mengambil pesan yang dipesan dari grup pesan lain, grup pesan pertama tetap tidak tersedia sampai pesan yang memblokir antrian diproses dengan sukses atau dipindahkan ke antrian huruf mati.

Selain itu, antrian FIFO memungkinkan jumlah pesan dalam penerbangan yang lebih rendah. Agar antrian FIFO Anda tidak diblokir oleh pesan, pastikan aplikasi Anda menangani pemrosesan pesan dengan benar.

Untuk informasi selengkapnya, lihat Kuota pesan Amazon SQS dan Bekerja dengan pesan Amazon SQS.

Masalah DLQ-redrive

Pelajari tentang masalah umum DLQ-redrive dan cara mengatasinya.

Topik

AccessDenied masalah izin

`AccessDenied`Kesalahan terjadi ketika redrive DLQ gagal karena entitas AWS Identity and Access Management (IAM) tidak memiliki izin yang diperlukan.

Contoh pesan kesalahan:

```
Failed to create redrive task. Error code: AccessDenied - Queue Permissions to Redrive.
```

Izin API berikut diperlukan untuk membuat permintaan redrive DLQ:

Untuk memulai redrive pesan:

- Izin antrian surat mati:
  - `sqs:StartMessageMoveTask`
  - `sqs:ReceiveMessage`
  - `sqs:DeleteMessage`
  - `sqs:GetQueueAttributes`
  - `kms:Decrypt`— Ketika antrian huruf mati atau antrian sumber asli dienkripsi.
- Izin antrian tujuan:
  - `sqs:SendMessage`
  - `kms:GenerateDataKey`— Saat antrian tujuan dienkripsi.
  - `kms:Decrypt` — Saat antrian tujuan dienkripsi.

Untuk membatalkan redrive pesan yang sedang berlangsung:

- Izin antrian surat mati:

- `sqs:CancelMessageMoveTask`
- `sqs:ReceiveMessage`
- `sqs:DeleteMessage`
- `sqs:GetQueueAttributes`
- `kms:Decrypt`— Ketika antrian huruf mati atau antrian sumber asli dienkripsi.

Untuk menampilkan status pemindahan pesan:

- Izin antrian surat mati:
  - `sqs:ListMessageMoveTasks`
  - `sqs:GetQueueAttributes`

NonExistentQueue kesalahan

`NonExistentQueue`Kesalahan terjadi ketika antrean sumber Amazon SQS tidak ada, atau dihapus. Periksa dan redrive ke antrian Amazon SQS yang ada.

Contoh pesan kesalahan:

```
Failed: AWS.SimpleQueueService.NonExistentQueue
```

CouldNotDetermineMessageKesalahan sumber

`CouldNotDetermineMessageSource`Kesalahan terjadi ketika Anda mencoba memulai redrive DLQ dengan skenario berikut:

- Pesan Amazon SQS dikirim langsung ke DLQ dengan API. `SendMessage`
- Pesan dari topik AWS Lambda atau fungsi Amazon Simple Notification Service (Amazon SNS) dengan DLQ yang dikonfigurasi.

Untuk mengatasi kesalahan ini, pilih Recrive ke tujuan kustom saat Anda memulai redrive. Kemudian, masukkan ARN antrian Amazon SQS untuk memindahkan semua pesan dari DLQ ke antrian tujuan.

Contoh pesan kesalahan:

```
Failed: CouldNotDetermineMessageSource
```

Memecahkan masalah pembatasan FIFO di Amazon SQS

Secara default, antrian FIFO mendukung 300 transaksi per detik, per tindakan API untuk, `SendMessage`, `ReceiveMessage`dan. `DeleteMessage` Permintaan lebih dari 300 TPS mendapatkan `ThrottlingException` kesalahan bahkan jika pesan dalam antrian tersedia. Untuk mengurangi ini, Anda dapat menggunakan metode berikut:

- Aktifkan throughput tinggi untuk antrian FIFO di Amazon SQS.
- Gunakan tindakan batch Amazon SQS API `SendMessageBatch``DeleteMessageBatch`, dan `ChangeMessageVisibilityBatch` untuk meningkatkan batas TPS hingga 3.000 pesan per detik per tindakan API, dan untuk mengurangi biaya. Untuk `ReceiveMessage` API, atur `MaxNumberofMessages` parameter untuk menerima hingga sepuluh pesan per transaksi. Untuk informasi selengkapnya, lihat Tindakan batch Amazon SQS.
- Untuk antrian FIFO dengan throughput tinggi, ikuti rekomendasi untuk mengoptimalkan pemanfaatan partisi. Kirim pesan dengan ID grup pesan yang sama dalam batch. Hapus pesan, atau ubah nilai batas waktu visibilitas pesan dalam batch dengan pegangan tanda terima dari permintaan API yang sama`ReceiveMessage`.
- Tingkatkan jumlah `MessageGroupId`nilai unik. Hal ini memungkinkan untuk distribusi merata di seluruh partisi antrian FIFO. Untuk informasi selengkapnya, lihat Menggunakan ID grup pesan Amazon SQS.

Untuk informasi selengkapnya, lihat Mengapa antrean Amazon SQS FIFO saya tidak menampilkan semua pesan atau pesan di grup pesan lain? dalam Panduan Pusat AWS Pengetahuan.

Memecahkan masalah pesan yang tidak dikembalikan untuk panggilan Amazon SQS API ReceiveMessage

Topik berikut mencakup penyebab paling umum mengapa pesan Amazon SQS mungkin tidak dikembalikan ke konsumen, dan cara memecahkan masalah mereka. Untuk informasi selengkapnya, lihat Mengapa saya tidak dapat menerima pesan dari antrian Amazon SQS saya? dalam Panduan Pusat AWS Pengetahuan.

Topik

- Antrian kosong
- Dalam batas penerbangan tercapai

- Penundaan pesan
- Pesan sedang dalam penerbangan
- Metode polling

Antrian kosong

Untuk menentukan apakah antrian kosong, gunakan polling panjang untuk memanggil API. `ReceiveMessage` Anda juga dapat menggunakan`ApproximateNumberOfMessagesVisible`,`ApproximateNumberOfMessagesNotVisib` dan `ApproximateNumberOfMessagesDelayed` CloudWatch metrik. Jika semua nilai metrik diatur ke 0 selama beberapa menit, antrian dianggap kosong.

Dalam batas penerbangan tercapai

Jika Anda menggunakan polling panjang dan jika antrian dalam batas penerbangan (20000 untuk FIFO, 120000 untuk standar secara default) dilanggar, Amazon SQS tidak akan menampilkan pesan kesalahan yang melebihi batas kuota.

Penundaan pesan

Jika antrean Amazon SQS dikonfigurasi sebagai antrean penundaan, atau pesan dikirim dengan pengatur waktu pesan, maka pesan tidak akan terlihat hingga waktu tunda berakhir. Untuk memverifikasi apakah antrian dikonfigurasi sebagai antrean penundaan, gunakan **`DelaySeconds`** atribut **`GetQueueAttributes`**API, atau dari konsol antrian di bawah Penundaan pengiriman. Periksa `ApproximateNumberOfMessagesDelayed` CloudWatch metrik untuk memahami apakah ada pesan yang tertunda.

Pesan sedang dalam penerbangan

Jika konsumen lain telah melakukan polling pesan, pesan akan dalam penerbangan atau tidak terlihat untuk periode batas waktu visibilitas. Jajak pendapat tambahan mungkin mengembalikan penerimaan kosong. Periksa CloudWatch metrik ApproximateNumberOfMessagesTerlihat untuk memahami jumlah pesan yang tersedia untuk diterima. Dalam kasus antrian FIFO, jika pesan dengan ID grup pesan sedang dalam penerbangan, maka tidak ada lagi pesan yang akan dikembalikan kecuali Anda menghapus pesan, atau menjadi terlihat. Ini karena pengurutan pesan dipertahankan pada tingkat grup pesan dalam antrian FIFO.

Metode polling

Jika Anda menggunakan polling singkat, (WaitTimeDetik adalah 0) Amazon SQS mengambil sampel subset servernya, dan mengembalikan pesan hanya dari server tersebut. Oleh karena itu, Anda mungkin tidak mendapatkan pesan bahkan jika mereka tersedia untuk diterima. Permintaan jajak pendapat berikutnya akan mengembalikan pesan.

Jika Anda menggunakan polling panjang, Amazon SQS polling semua server dan mengirimkan respons setelah mengumpulkan setidaknya satu pesan yang tersedia, dan hingga jumlah maksimum yang ditentukan. Jika nilai untuk ReceiveMessage WaitTimeDetik terlalu rendah, Anda mungkin tidak menerima semua pesan yang tersedia.

Memecahkan masalah kesalahan jaringan Amazon SQS

Topik berikut mencakup penyebab paling umum untuk masalah jaringan di Amazon SQS, dan cara memecahkan masalah mereka.

Topik

ETIMEOUT error

ETIMEOUTKesalahan terjadi ketika klien tidak dapat membuat koneksi TCP ke titik akhir Amazon SQS.

Pemecahan masalah:

- Periksa koneksi jaringan

  Uji koneksi jaringan Anda ke Amazon SQS dengan menjalankan perintah seperti. `telnet`

  Example: telnet sqs.us-east-1.amazonaws.com 443
- Periksa pengaturan jaringan
  - Pastikan aturan firewall lokal, rute, dan daftar kontrol akses (ACL) mengizinkan lalu lintas di port yang Anda gunakan.
  - Aturan keluar (jalan keluar) grup keamanan harus mengizinkan lalu lintas ke port 80 atau 443.
  - Aturan keluar (jalan keluar) ACL jaringan harus mengizinkan lalu lintas ke port TCP 80 atau 443.

  - Aturan inbound (ingress) ACL jaringan harus mengizinkan lalu lintas pada port TCP 1024-65535.
  - Instans Amazon Elastic Compute Cloud (Amazon EC2) yang terhubung ke internet publik harus memiliki konektivitas internet.
- Titik akhir Amazon Virtual Private Cloud (Amazon VPC)

  Jika Anda mengakses Amazon SQS melalui titik akhir VPC Amazon, grup keamanan titik akhir harus mengizinkan lalu lintas masuk ke grup keamanan klien di port 443. ACL jaringan yang terkait dengan subnet titik akhir VPC harus memiliki konfigurasi ini:

  - Aturan keluar (jalan keluar) ACL jaringan harus mengizinkan lalu lintas pada port TCP 1024-65535 (port sementara).
  - Aturan inbound (ingress) ACL jaringan harus mengizinkan lalu lintas di port 443.

Selain itu, kebijakan titik akhir Amazon SQS VPC AWS Identity and Access Management (IAM) harus mengizinkan akses. Contoh kebijakan titik akhir VPC berikut menetapkan bahwa pengguna IAM diizinkan untuk mengirim pesan ke *MyUser*antrean Amazon SQS. *MyQueue* Tindakan lain, pengguna IAM, dan sumber daya Amazon SQS ditolak aksesnya melalui titik akhir VPC.

```
{
    "Statement": [{
        "Action": ["sqs:SendMessage"],
        "Effect": "Allow",
        "Resource": "arn:aws:sqs:us-east-2:123456789012:MyQueue",
        "Principal": {
            "AWS": "arn:aws:iam:123456789012:user/MyUser"
        }
    }]
}
```

UnknownHostException error

UnknownHostExceptionKesalahan terjadi ketika alamat IP host tidak dapat ditentukan.

Pemecahan masalah:

Gunakan nslookup utilitas untuk mengembalikan alamat IP yang terkait dengan nama host:

- Windows and Linux OS

```
nslookup sqs.<region>.amazonaws.com
```

- AWS CLI atau SDK untuk titik akhir lama Python:

```
nslookup <region>.queue.amazonaws.com
```

Jika Anda menerima output yang gagal, ikuti instruksi di Bagaimana cara kerja DNS dan bagaimana cara memecahkan masalah kegagalan DNS sebagian atau intermiten? dalam Panduan Pusat AWS Pengetahuan.

Jika Anda menerima output yang valid, maka kemungkinan akan menjadi masalah tingkat aplikasi. Untuk mengatasi masalah tingkat aplikasi, coba metode berikut:

- Mulai ulang aplikasi Anda.
- Konfirmasikan bahwa aplikasi Java Anda tidak memiliki cache DNS yang buruk. Jika memungkinkan, konfigurasikan aplikasi Anda untuk mematuhi DNS TTL. Untuk informasi selengkapnya, lihat Mengatur TTL JVM untuk pencarian nama DNS.

Untuk informasi tambahan tentang cara memecahkan masalah kesalahan jaringan, lihat Bagaimana cara mengatasi kesalahan koneksi “ETIMEOUT” dan “Pengecualian” Amazon SQS? UnknownHost dalam Panduan Pusat AWS Pengetahuan.

Memecahkan masalah antrian Amazon Simple Queue Service menggunakan AWS X-Ray

AWS X-Ray mengumpulkan data tentang permintaan yang disajikan aplikasi Anda dan memungkinkan Anda melihat dan memfilter data untuk mengidentifikasi potensi masalah dan peluang untuk pengoptimalan. Untuk setiap permintaan yang dilacak ke aplikasi Anda, Anda dapat melihat informasi terperinci tentang permintaan, respons, dan panggilan yang dilakukan aplikasi Anda ke AWS sumber daya hilir, layanan mikro, database, dan API web HTTP.

Untuk mengirim header AWS X-Ray jejak melalui Amazon SQS, Anda dapat melakukan salah satu hal berikut:

- Gunakan header `X-Amzn-Trace-Id` tracing.
- Gunakan atribut sistem `AWSTraceHeader` pesan.

Untuk mengumpulkan data tentang kesalahan dan latensi, Anda harus instrumen `AmazonSQS`klien menggunakan AWS X-Ray SDK.

Anda dapat menggunakan AWS X-Ray konsol untuk melihat peta koneksi antara Amazon SQS dan layanan lain yang digunakan aplikasi Anda. Anda juga dapat menggunakan konsol tersebut untuk melihat metrik seperti tingkat latensi dan kegagalan rata-rata. Untuk informasi selengkapnya, lihat Amazon SQS dan AWS X-Ray di Panduan AWS X-Ray Pengembang.

Keamanan di Amazon SQS

Bagian ini menyediakan informasi tentang keamanan Amazon SQS, otentikasi dan kontrol akses, dan Bahasa Kebijakan Akses Amazon SQS.

Topik

Perlindungan data di Amazon SQS

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data di Amazon Simple Queue Service. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Lihat informasi yang lebih lengkap tentang privasi data dalam Pertanyaan Umum Privasi Data. Lihat informasi tentang perlindungan data di Eropa di pos blog Model Tanggung Jawab Bersama dan GDPR AWS di Blog Keamanan AWS .

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensil dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan sumber daya. AWS Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail.

- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-2 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi yang lebih lengkap tentang titik akhir FIPS yang tersedia di Standar Pemrosesan Informasi Federal (FIPS) 140-2.

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan Amazon SQS atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau AWS SDK. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan atau log diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

Bagian berikut memberikan informasi tentang perlindungan data di Amazon SQS.

Topik

Enkripsi data di Amazon SQS

Perlindungan data mengacu pada melindungi data saat dalam perjalanan (saat bepergian ke dan dari Amazon SQS) dan saat istirahat (saat disimpan di disk di pusat data Amazon SQS). Anda dapat melindungi data saat transit menggunakan Secure Socket Layer (SSL) atau enkripsi di sisi klien. Secara default, Amazon SQS menyimpan pesan dan file menggunakan enkripsi disk. Anda dapat melindungi data saat istirahat dengan meminta Amazon SQS untuk mengenkripsi pesan Anda sebelum menyimpannya ke sistem file terenkripsi di pusat datanya. Amazon SQS merekomendasikan penggunaan SSE untuk enkripsi data yang dioptimalkan.

Topik

Enkripsi saat istirahat di Amazon SQS

Server-side encryption (SSE) memungkinkan Anda mengirimkan data sensitif dalam antrian terenkripsi. SSE melindungi isi pesan dalam antrian menggunakan kunci enkripsi yang dikelola SQS (SSE-SQS) atau kunci yang dikelola di (SSE-KMS). AWS Key Management Service Untuk informasi tentang mengelola SSE menggunakan AWS Management Console, lihat berikut ini:

- Mengkonfigurasi SSE-SQS untuk antrian (konsol)
- Mengkonfigurasi SSE-KMS untuk antrian (konsol)

Untuk informasi tentang mengelola SSE menggunakan AWS SDK for Java (dan`CreateQueue`,`SetQueueAttributes`, dan `GetQueueAttributes` tindakan), lihat contoh berikut:

- Menggunakan enkripsi sisi server dengan antrian Amazon SQS
- Mengkonfigurasi izin KMS untuk Layanan AWS

SSE mengenkripsi pesan segera setelah Amazon SQS menerimanya. Pesan disimpan dalam bentuk terenkripsi dan Amazon SQS mendekripsi pesan hanya ketika dikirim ke konsumen yang berwenang.

> **Important**
>
> Semua permintaan untuk antrian dengan SSE diaktifkan harus menggunakan HTTPS dan Signature Version 4.
> Antrian terenkripsi yang menggunakan kunci default (kunci KMS AWS terkelola untuk Amazon SQS) tidak dapat menjalankan fungsi Lambda secara berbeda. Akun AWS
> Beberapa fitur AWS layanan yang dapat mengirim pemberitahuan ke Amazon SQS menggunakan AWS Security Token Service `AssumeRole` tindakan kompatibel dengan SSE tetapi hanya berfungsi dengan antrian standar:
>
> - Kait Siklus Hidup Auto Scaling
> - AWS Lambda Antrian Surat Mati
>
> Untuk informasi tentang kompatibilitas layanan lain dengan antrian terenkripsi, lihat Konfigurasikan izin KMS untuk layanan AWS dan dokumentasi layanan Anda.

AWS KMS menggabungkan perangkat keras dan perangkat lunak yang aman dan sangat tersedia untuk menyediakan sistem manajemen kunci yang diskalakan untuk cloud. Saat Anda menggunakan Amazon SQS AWS KMS, kunci data yang mengenkripsi data pesan Anda juga dienkripsi dan disimpan dengan data yang dilindunginya.

Berikut ini adalah manfaat menggunakan AWS KMS:

- Anda dapat membuat dan mengelola AWS KMS keysdiri sendiri.
- Anda juga dapat menggunakan kunci KMS AWS terkelola untuk Amazon SQS, yang unik untuk setiap akun dan wilayah.
- Standar AWS KMS keamanan dapat membantu Anda memenuhi persyaratan kepatuhan terkait enkripsi.

Untuk informasi lebih lanjut, lihat Apa yang dimaksud AWS Key Management Service? dalam Panduan Developer AWS Key Management Service .

Topik

- Lingkup enkripsi
- Istilah kunci

Lingkup enkripsi

SSE mengenkripsi isi pesan dalam antrian Amazon SQS.

SSE tidak mengenkripsi berikut ini:

- Metadata antrian (nama antrian dan atribut)
- Metadata pesan (ID pesan, stempel waktu, dan atribut)
- Metrik per antrian

Mengenkripsi pesan membuat isinya tidak tersedia untuk pengguna yang tidak sah atau anonim. Dengan SSE diaktifkan, anonim `SendMessage` dan `ReceiveMessage` permintaan ke antrian terenkripsi akan ditolak. Praktik terbaik keamanan Amazon SQS merekomendasikan agar tidak menggunakan permintaan anonim. Jika Anda ingin mengirim permintaan anonim ke antrian Amazon SQS, pastikan Anda menonaktifkan SSE. Ini tidak memengaruhi fungsi normal Amazon SQS:

- Pesan dienkripsi hanya jika dikirim setelah enkripsi antrian diaktifkan. Amazon SQS tidak mengenkripsi pesan yang di-backlog.
- Setiap pesan terenkripsi tetap dienkripsi bahkan jika enkripsi antreannya dinonaktifkan.

Memindahkan pesan ke antrian huruf mati tidak memengaruhi enkripsi:

- Saat Amazon SQS memindahkan pesan dari antrian sumber terenkripsi ke antrian huruf mati yang tidak terenkripsi, pesan tetap terenkripsi.
- Saat Amazon SQS memindahkan pesan dari antrian sumber yang tidak terenkripsi ke antrian huruf mati terenkripsi, pesan tetap tidak terenkripsi.

Istilah kunci

Istilah kunci berikut ini dapat membantu Anda lebih memahami fungsionalitas SSE. Untuk deskripsi mendetail, lihat Referensi API Layanan Antrian Sederhana Amazon.

Kunci data

Kunci (DEK) bertanggung jawab untuk mengenkripsi konten pesan Amazon SQS.

Untuk informasi selengkapnya, lihat Kunci Data di Panduan AWS Key Management Service Pengembang di Panduan AWS Encryption SDK Pengembang.

Periode penggunaan kembali kunci data

Lamanya waktu, dalam hitungan detik, Amazon SQS dapat menggunakan kembali kunci data untuk mengenkripsi atau mendekripsi pesan sebelum menelepon lagi. AWS KMS Bilangan bulat yang mewakili detik, antara 60 detik (1 menit) dan 86.400 detik (24 jam). Defaultnya adalah 300 (5 menit). Untuk informasi selengkapnya, lihat Memahami periode penggunaan kembali kunci data.

> **Note**
>
> Jika tidak dapat menjangkau AWS KMS, Amazon SQS terus menggunakan kunci data yang di-cache hingga koneksi dibuat kembali.

ID kunci KMS

Alias, alias ARN, ID kunci, atau ARN kunci dari kunci KMS AWS terkelola atau kunci KMS kustom—di akun Anda atau di akun lain. Sementara alias kunci KMS AWS terkelola untuk Amazon SQS

`alias/aws/sqs` selalu, alias kunci KMS kustom dapat, misalnya,. `alias/MyAlias` Anda dapat menggunakan kunci KMS ini untuk melindungi pesan di antrian Amazon SQS.

> **Note**
>
> Ingatlah hal-hal berikut ini:
>
> - Jika Anda tidak menentukan kunci KMS kustom, Amazon SQS menggunakan kunci KMS AWS terkelola untuk Amazon SQS.
> - Pertama kali Anda menggunakan AWS Management Console untuk menentukan kunci KMS AWS terkelola untuk Amazon SQS untuk antrean AWS KMS , membuat AWS kunci KMS terkelola untuk Amazon SQS.
> - Atau, saat pertama kali Anda menggunakan `SendMessageBatch` tindakan `SendMessage` atau pada antrian dengan SSE diaktifkan, AWS KMS membuat kunci KMS AWS terkelola untuk Amazon SQS.

Anda dapat membuat kunci KMS, menentukan kebijakan yang mengontrol bagaimana kunci KMS dapat digunakan, dan mengaudit penggunaan kunci KMS menggunakan bagian Kunci terkelola Pelanggan di AWS KMS konsol atau tindakan. `CreateKey` AWS KMS Untuk informasi selengkapnya, lihat kunci KMS dan Membuat Kunci di Panduan AWS Key Management Service Pengembang. Untuk lebih banyak contoh pengidentifikasi kunci KMS, lihat KeyIddi Referensi AWS Key Management Service API. Untuk informasi tentang menemukan pengidentifikasi kunci KMS, lihat Menemukan ID Kunci dan ARN di Panduan Pengembang.AWS Key Management Service

> **Important**
>
> Ada biaya tambahan untuk penggunaan AWS KMS. Untuk informasi selengkapnya, lihat Memperkirakan biaya AWS KMS dan Harga AWS Key Management Service.

Enkripsi Amplop

Keamanan data terenkripsi Anda sebagian bergantung pada perlindungan kunci data yang dapat mendekripsi itu. Amazon SQS menggunakan kunci KMS untuk mengenkripsi kunci data dan kemudian kunci data terenkripsi disimpan dengan pesan terenkripsi. Praktik menggunakan kunci KMS untuk mengenkripsi kunci data ini dikenal sebagai enkripsi amplop.

Untuk informasi selengkapnya, lihat Enkripsi envelope di Panduan Developer AWS Encryption SDK .

Manajemen Kunci Amazon SQS

Amazon SQS terintegrasi dengan AWS Key Management Service (KMS) untuk mengelola kunci KMS untuk enkripsi sisi server (SSE). Lihat Enkripsi saat istirahat di Amazon SQS untuk informasi SSE dan definisi manajemen kunci. Amazon SQS menggunakan kunci KMS untuk memvalidasi dan mengamankan kunci data yang mengenkripsi dan mendekripsi pesan. Bagian berikut memberikan informasi tentang bekerja dengan kunci KMS dan kunci data di layanan Amazon SQS.

Topik

- Mengonfigurasi izin AWS KMS
- Memahami periode penggunaan kembali kunci data
- Memperkirakan biaya AWS KMS
- AWS KMS kesalahan

Mengonfigurasi izin AWS KMS

Setiap kunci KMS harus memiliki kebijakan kunci. Perhatikan bahwa Anda tidak dapat mengubah kebijakan kunci kunci KMS AWS terkelola untuk Amazon SQS. Kebijakan untuk kunci KMS ini mencakup izin untuk semua prinsipal di akun (yang diizinkan untuk menggunakan Amazon SQS) untuk menggunakan antrian terenkripsi.

Untuk kunci KMS yang dikelola pelanggan, Anda harus mengonfigurasi kebijakan kunci untuk menambahkan izin bagi setiap produsen dan konsumen antrian. Untuk melakukan ini, Anda memberi nama produsen dan konsumen sebagai pengguna dalam kebijakan kunci KMS. Untuk informasi selengkapnya tentang AWS KMS izin, lihat referensi AWS KMS sumber daya dan operasi atau izin AWS KMS API di Panduan AWS Key Management Service Pengembang.

Atau, Anda dapat menentukan izin yang diperlukan dalam kebijakan IAM yang ditetapkan ke prinsipal yang menghasilkan dan menggunakan pesan terenkripsi. Untuk informasi selengkapnya, lihat Menggunakan Kebijakan IAM dengan AWS KMS Panduan AWS Key Management Service Pengembang.

> **Note**
>
> Meskipun Anda dapat mengonfigurasi izin global untuk mengirim dan menerima dari Amazon SQS AWS KMS , memerlukan penamaan ARN lengkap kunci KMS secara eksplisit di wilayah tertentu di bagian kebijakan IAM. `Resource`

Konfigurasikan izin KMS untuk layanan AWS

Beberapa AWS layanan bertindak sebagai sumber acara yang dapat mengirim acara ke antrian Amazon SQS. Agar sumber peristiwa ini dapat bekerja dengan antrian terenkripsi, Anda harus membuat kunci KMS yang dikelola pelanggan dan menambahkan izin dalam kebijakan kunci agar layanan dapat menggunakan metode API yang diperlukan. AWS KMS Lakukan langkah-langkah berikut untuk mengonfigurasi izin.

> **Warning**
>
> Saat mengubah kunci KMS untuk mengenkripsi pesan Amazon SQS Anda, ketahuilah bahwa pesan yang ada yang dienkripsi dengan kunci KMS lama akan tetap dienkripsi dengan kunci itu. Untuk mendekripsi pesan ini, Anda harus menyimpan kunci KMS lama dan memastikan bahwa kebijakan utamanya memberikan Amazon SQS izin untuk dan. `kms:Decrypt` `kms:GenerateDataKey` Setelah memperbarui ke kunci KMS baru untuk mengenkripsi pesan baru, pastikan semua pesan yang ada dienkripsi dengan kunci KMS lama diproses dan dihapus dari antrian sebelum menghapus atau menonaktifkan kunci KMS lama.

1. Buat kunci KMS yang dikelola pelanggan. Untuk informasi selengkapnya, lihat Membuat Kunci di Panduan Developer AWS Key Management Service .
2. Untuk mengizinkan sumber peristiwa AWS layanan menggunakan metode `kms:GenerateDataKey` dan `kms:Decrypt` API, tambahkan pernyataan berikut ke kebijakan kunci KMS.

```
{
   "Version": "2012-10-17",
      "Statement": [{
         "Effect": "Allow",
         "Principal": {
            "Service": "service.amazonaws.com"
         },
```

```
          "Action": [
             "kms:GenerateDataKey",
             "kms:Decrypt"
          ],
          "Resource": "*"
       }]
}
```

Ganti “layanan” pada contoh di atas dengan nama Layanan dari sumber acara. Sumber acara termasuk layanan berikut.

Sumber peristiwa	Nama layanan
CloudWatch Acara Amazon	`events.amazonaws.com`
Pemberitahuan acara Amazon S3	`s3.amazonaws.com`
Langganan topik Amazon SNS	`sns.amazonaws.com`

3. Konfigurasikan antrian SSE yang ada menggunakan ARN kunci KMS Anda.
4. Berikan ARN dari antrian terenkripsi ke sumber acara.

Konfigurasikan AWS KMS izin untuk produsen

Ketika periode penggunaan kembali kunci data berakhir, panggilan berikutnya produsen ke `SendMessage` atau `SendMessageBatch` juga memicu panggilan ke dan. `kms:GenerateDataKey` `kms:Decrypt` Panggilan `kms:Decrypt` untuk memverifikasi integritas kunci data baru sebelum menggunakannya. Oleh karena itu, produsen harus memiliki `kms:GenerateDataKey` dan `kms:Decrypt` izin untuk kunci KMS.

Tambahkan pernyataan berikut ke kebijakan IAM produsen. Ingatlah untuk menggunakan nilai ARN yang benar untuk sumber daya kunci dan sumber daya antrian.

```
{
   "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"
      }]
}
```

Konfigurasikan AWS KMS izin untuk konsumen

Ketika periode penggunaan kembali kunci data berakhir, panggilan konsumen berikutnya `ReceiveMessage` juga memicu panggilan kekms:Decrypt, untuk memverifikasi integritas kunci data baru sebelum menggunakannya. Oleh karena itu, konsumen harus memiliki `kms:Decrypt` izin untuk kunci KMS apa pun yang digunakan untuk mengenkripsi pesan dalam antrian yang ditentukan. Jika antrian bertindak sebagai antrian huruf mati, konsumen juga harus memiliki `kms:Decrypt` izin untuk setiap kunci KMS yang digunakan untuk mengenkripsi pesan dalam antrian sumber. Tambahkan pernyataan berikut ke kebijakan IAM konsumen. Ingatlah untuk menggunakan nilai ARN yang benar untuk sumber daya kunci dan sumber daya antrian.

```
{
   "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"
      }]
}
```

Konfigurasikan AWS KMS izin dengan perlindungan wakil yang membingungkan

Ketika prinsipal dalam pernyataan kebijakan kunci adalah prinsipal AWS layanan, Anda dapat menggunakan `aws:SourceArn`atau kunci kondisi `aws:SourceAccount`global untuk melindungi dari skenario wakil yang membingungkan. Untuk menggunakan kunci kondisi ini, tetapkan nilai ke Amazon Resource Name (ARN) dari sumber daya yang sedang dienkripsi. Jika Anda tidak tahu ARN sumber daya, gunakan `aws:SourceAccount` sebagai gantinya.

Dalam kebijakan kunci KMS ini, sumber daya tertentu dari layanan yang dimiliki oleh akun `111122223333` diizinkan untuk memanggil KMS `Decrypt` dan `GenerateDataKey` tindakan, yang terjadi selama penggunaan SSE 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"
    ]
   }
  }
 }]
}
```

Saat menggunakan antrian Amazon SQS yang diaktifkan SSE, layanan berikut mendukung: `aws:SourceArn`

- Amazon SNS
- Amazon S3
- CloudWatch Acara
- AWS Lambda

- CodeBuild
- Amazon Connect Customer Profiles
- AWS Auto Scaling
- Amazon Chime

Memahami periode penggunaan kembali kunci data

Periode penggunaan kembali kunci data menentukan durasi maksimum Amazon SQS untuk menggunakan kembali kunci data yang sama. Ketika periode penggunaan kembali kunci data berakhir, Amazon SQS menghasilkan kunci data baru. Perhatikan pedoman berikut tentang periode penggunaan kembali.

- Periode penggunaan kembali yang lebih pendek memberikan keamanan yang lebih baik tetapi menghasilkan lebih banyak panggilan ke AWS KMS, yang mungkin dikenakan biaya di luar Tingkat Gratis.
- Meskipun kunci data di-cache secara terpisah untuk enkripsi dan dekripsi, periode penggunaan kembali berlaku untuk kedua salinan kunci data.
- Ketika periode penggunaan kembali kunci data berakhir, panggilan berikutnya ke `SendMessage` atau `SendMessageBatch` biasanya memicu panggilan ke AWS KMS `GenerateDataKey` metode untuk mendapatkan kunci data baru. Juga, panggilan berikutnya ke `SendMessage` dan masing-masing `ReceiveMessage` akan memicu panggilan AWS KMS `Decrypt` untuk memverifikasi integritas kunci data sebelum menggunakannya.
- Prinsipal (Akun AWS atau pengguna) tidak berbagi kunci data (pesan yang dikirim oleh prinsipal unik selalu mendapatkan kunci data unik). Oleh karena itu, volume panggilan ke AWS KMS adalah kelipatan dari jumlah prinsipal unik yang digunakan selama periode penggunaan kembali kunci data.

Memperkirakan biaya AWS KMS

Untuk memprediksi biaya dan lebih memahami AWS tagihan Anda, Anda mungkin ingin tahu seberapa sering Amazon SQS menggunakan kunci KMS Anda.

> **Note**
>
> Meskipun rumus berikut dapat memberi Anda gambaran yang sangat baik tentang biaya yang diharapkan, biaya sebenarnya mungkin lebih tinggi karena sifat terdistribusi Amazon SQS.

Untuk menghitung jumlah permintaan API (R) per antrian, gunakan rumus berikut:

```
R = (B / D) * (2 * P + C)
```

B adalah periode penagihan (dalam detik).

Dadalah periode penggunaan kembali kunci data (dalam detik).

Padalah jumlah prinsipal produksi yang mengirim ke antrian Amazon SQS.

Cadalah jumlah prinsipal konsumsi yang menerima dari antrian Amazon SQS.

> **Important**
>
> Secara umum, prinsipal produksi dikenakan biaya dua kali lipat biaya konsumsi prinsipal. Untuk informasi selengkapnya, lihat Memahami periode penggunaan kembali kunci data. Jika produsen dan konsumen memiliki pengguna yang berbeda, biayanya meningkat.

Berikut ini adalah contoh perhitungan. Untuk informasi harga sebenranya, lihat Harga AWS Key Management Service.

Contoh 1: Menghitung jumlah panggilan AWS KMS API untuk 2 prinsipal dan 1 antrian

Contoh ini mengasumsikan sebagai berikut:

- Periode penagihan adalah 1-31 Januari (2.678.400 detik).
- Periode penggunaan kembali kunci data diatur ke 5 menit (300 detik).
- Ada 1 antrian.
- Ada 1 pokok produksi dan 1 pokok konsumsi.

```
(2,678,400 / 300) * (2 * 1 + 1) = 26,784
```

Contoh 2: Menghitung jumlah panggilan AWS KMS API untuk beberapa produsen dan konsumen dan 2 antrian

Contoh ini mengasumsikan sebagai berikut:

- Periode penagihan adalah 1-28 Februari (2.419.200 detik).
- Periode penggunaan kembali kunci data diatur ke 24 jam (86.400 detik).
- Ada 2 antrian.
- Antrian pertama memiliki 3 prinsip produksi dan 1 pokok konsumsi.
- Antrian kedua memiliki 5 prinsip produksi dan 2 prinsip konsumsi.

```
(2,419,200 / 86,400 * (2 * 3 + 1)) + (2,419,200 / 86,400 * (2 * 5 + 2)) = 532
```

AWS KMS kesalahan

Saat Anda bekerja dengan Amazon SQS dan AWS KMS, Anda mungkin mengalami kesalahan. Referensi berikut menjelaskan kesalahan dan kemungkinan solusi pemecahan masalah.

- AWS KMS Kesalahan umum
- AWS KMS Mendekripsi kesalahan
- AWS KMS GenerateDataKey kesalahan

Privasi lalu lintas internetwork di Amazon SQS

Titik akhir Amazon Virtual Private Cloud (Amazon VPC) untuk Amazon SQS adalah entitas logis dalam VPC yang memungkinkan konektivitas hanya ke Amazon SQS. VPC merutekan permintaan ke Amazon SQS dan merutekan respons kembali ke VPC. Bagian berikut ini memberikan informasi tentang bekerja dengan VPC endpoint dan membuat kebijakan VPC endpoint.

Topik

- Titik akhir Amazon Virtual Private Cloud untuk Amazon SQS
- Membuat kebijakan titik akhir Amazon VPC untuk Amazon SQS

Titik akhir Amazon Virtual Private Cloud untuk Amazon SQS

Jika Anda menggunakan Amazon VPC untuk meng-host AWS sumber daya, Anda dapat membuat sambungan antara VPC dan Amazon SQS. Anda dapat menggunakan koneksi ini untuk mengirim pesan ke antrian Amazon SQS Anda tanpa melintasi internet publik.

Amazon VPC memungkinkan Anda meluncurkan AWS sumber daya di jaringan virtual khusus. Anda dapat menggunakan VPC untuk mengendalikan pengaturan jaringan, seperti rentang alamat IP, subnet, tabel rute, dan gateway jaringan. Untuk informasi lebih lanjut tentang Amazon VPC, lihat Panduan Pengguna Amazon VPC.

Untuk menghubungkan VPC Anda ke Amazon SQS, Anda harus terlebih dahulu menentukan titik akhir VPC antarmuka, yang memungkinkan Anda menghubungkan VPC ke layanan lain. AWS Titik akhir menyediakan konektivitas yang andal dan dapat diskalakan ke Amazon SQS tanpa memerlukan gateway internet, instance terjemahan alamat jaringan (NAT), atau koneksi VPN. Untuk informasi selengkapnya, lihat Tutorial: Mengirim pesan ke antrian Amazon SQS dari Amazon Virtual Private Cloud dan Contoh 5: Tolak akses jika bukan dari titik akhir VPC dalam panduan ini dan Titik Akhir VPC Antarmuka (AWS PrivateLink) di Panduan Pengguna Amazon VPC.

> **Important**
>
> - Anda dapat menggunakan Amazon Virtual Private Cloud hanya dengan titik akhir HTTPS Amazon SQS.
> - Saat mengonfigurasi Amazon SQS untuk mengirim pesan dari Amazon VPC, Anda harus mengaktifkan DNS pribadi dan menentukan titik akhir dalam format. `sqs.us-east-2.amazonaws.com`
> - DNS pribadi tidak mendukung titik akhir lama seperti atau. `queue.amazonaws.com` `us-east-2.queue.amazonaws.com`

Membuat kebijakan titik akhir Amazon VPC untuk Amazon SQS

Anda dapat membuat kebijakan untuk titik akhir VPC Amazon untuk Amazon SQS yang Anda tentukan sebagai berikut:

- Prinsipal yang dapat melakukan tindakan.
- Tindakan yang dapat dilakukan.

- Sumber daya yang menjadi target tindakan.

Untuk informasi selengkapnya, lihat Mengontrol Akses ke Layanan dengan Titik Akhir VPC di Panduan Pengguna Amazon VPC

Contoh kebijakan titik akhir VPC berikut menetapkan bahwa pengguna `MyUser` diizinkan mengirim pesan ke antrian 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"
      }
   }]
}
```

Hal berikut ini ditolak:

- Tindakan API Amazon SQS lainnya, seperti `sqs:CreateQueue` dan. `sqs:DeleteQueue`
- Pengguna dan aturan lain yang mencoba menggunakan titik akhir VPC ini.
- `MyUser`mengirim pesan ke antrian Amazon SQS yang berbeda.

> **Note**
>
> Pengguna masih dapat menggunakan tindakan Amazon SQS API lainnya dari luar VPC. Untuk informasi selengkapnya, lihat Contoh 5: Tolak akses jika bukan dari titik akhir VPC.

Manajemen identitas dan akses di Amazon SQS

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diotorisasi (memiliki izin) untuk menggunakan sumber daya Amazon SQS. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan di Amazon SQS.

Pengguna layanan - Jika Anda menggunakan layanan Amazon SQS untuk melakukan pekerjaan Anda, administrator Anda memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak fitur Amazon SQS untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur di Amazon SQS, lihat. Memecahkan masalah identitas dan akses Amazon Simple Queue Service

Administrator layanan - Jika Anda bertanggung jawab atas sumber daya Amazon SQS di perusahaan Anda, Anda mungkin memiliki akses penuh ke Amazon SQS. Tugas Anda adalah menentukan fitur dan sumber daya Amazon SQS mana yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep Basic IAM. Untuk mempelajari selengkapnya tentang bagaimana perusahaan Anda dapat menggunakan IAM dengan Amazon SQS, lihat. Bagaimana Amazon Simple Queue Service bekerja dengan IAM

Administrator IAM - Jika Anda administrator IAM, Anda mungkin ingin mempelajari detail tentang cara menulis kebijakan untuk mengelola akses ke Amazon SQS. Untuk melihat contoh kebijakan berbasis identitas Amazon SQS yang dapat Anda gunakan di IAM, lihat. Praktik terbaik kebijakan

Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensi identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk AWS sebagai identitas federasi dengan menggunakan kredensi yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (IAM Identity Center), autentikasi masuk tunggal perusahaan Anda, dan kredensi Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas terfederasi, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal AWS akses. Untuk informasi selengkapnya tentang masuk AWS, lihat Cara masuk ke Panduan AWS Sign-In Pengguna Anda Akun AWS.

Jika Anda mengakses AWS secara terprogram, AWS sediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensil Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Untuk informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, lihat Menandatangani permintaan AWS API di Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat Autentikasi multi-faktor dalam Panduan Pengguna AWS IAM Identity Center dan Menggunakan autentikasi multi-faktor (MFA) dalam AWS dalam Panduan Pengguna IAM.

Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihat Tugas yang memerlukan kredensial pengguna root dalam Panduan Pengguna IAM.

Identitas gabungan

Sebagai praktik terbaik, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensi sementara.

Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, direktori Pusat Identitas AWS Directory Service, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensi yang disediakan melalui sumber identitas. Ketika identitas federasi mengakses Akun AWS, mereka mengambil peran, dan peran memberikan kredensi sementara.

Untuk manajemen akses terpusat, kami sarankan Anda menggunakan AWS IAM Identity Center. Anda dapat membuat pengguna dan grup di Pusat Identitas IAM, atau Anda dapat menghubungkan dan menyinkronkan ke sekumpulan pengguna dan grup di sumber identitas Anda sendiri untuk

digunakan di semua aplikasi Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, lihat Apakah itu Pusat Identitas IAM? dalam Panduan Pengguna AWS IAM Identity Center .

Pengguna dan grup IAM

Pengguna IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, kami merekomendasikan untuk mengandalkan kredensial sementara, bukan membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan tertentu yang memerlukan kredensial jangka panjang dengan pengguna IAM, kami merekomendasikan Anda merotasi kunci akses. Untuk informasi selengkapnya, lihat Merotasi kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensial jangka panjang dalam Panduan Pengguna IAM.

Grup IAM adalah identitas yang menentukan sekumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin bagi beberapa pengguna sekaligus. Grup mempermudah manajemen izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat memiliki grup yang bernama IAMAdmins dan memberikan izin ke grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari selengkapnya, lihat Kapan harus membuat pengguna IAM (bukan peran) dalam Panduan Pengguna IAM.

Peran IAM

Peran IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Anda dapat mengambil peran IAM untuk sementara AWS Management Console dengan beralih peran. Anda dapat mengambil peran dengan memanggil operasi AWS CLI atau AWS API atau dengan menggunakan URL kustom. Untuk informasi selengkapnya tentang cara menggunakan peran, lihat Menggunakan peran IAM dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Ketika identitas terfederasi mengautentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk

informasi tentang peran untuk federasi, lihat Membuat peran untuk Penyedia Identitas pihak ketiga dalam Panduan Pengguna IAM. Jika menggunakan Pusat Identitas IAM, Anda harus mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM akan mengorelasikan set izin ke peran dalam IAM. Untuk informasi tentang set izin, lihat Set izin dalam Panduan Pengguna AWS IAM Identity Center .

- Izin pengguna IAM sementara – Pengguna atau peran IAM dapat mengambil peran IAM guna mendapatkan berbagai izin secara sementara untuk tugas tertentu.
- Akses lintas akun – Anda dapat menggunakan peran IAM untuk mengizinkan seseorang (prinsipal tepercaya) di akun lain untuk mengakses sumber daya di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM.
- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Sebagai contoh, ketika Anda memanggil suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.
  - Sesi akses teruskan (FAS) — Saat Anda menggunakan pengguna IAM atau peran untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.
  - Peran layanan – Peran layanan adalah peran IAM yang dijalankan oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Membuat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.
  - Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan peran IAM untuk mengelola kredensi sementara untuk aplikasi yang berjalan pada instans EC2 dan membuat atau permintaan API. AWS CLI AWS Cara ini lebih dianjurkan daripada menyimpan kunci akses dalam instans EC2. Untuk menetapkan AWS peran ke instans EC2 dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instance yang dilampirkan ke instance. Profil instans berisi peran dan memungkinkan program yang berjalan di instans EC2 mendapatkan kredensial sementara. Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan dalam instans Amazon EC2 dalam Panduan Pengguna IAM.

Untuk mempelajari apakah kita harus menggunakan peran IAM atau pengguna IAM, lihat Kapan harus membuat peran IAM (bukan pengguna) dalam Panduan Pengguna IAM.

Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal (pengguna, pengguna root, atau sesi peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, lihat Gambaran umum kebijakan JSON dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasinya. Misalnya, anggaplah Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut bisa mendapatkan informasi peran dari AWS Management Console, API AWS CLI, atau AWS API.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan yang dikelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan terkelola adalah kebijakan mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran dalam. Akun AWS Kebijakan AWS terkelola mencakup kebijakan terkelola dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan yang dikelola atau kebijakan inline, lihat Memilih antara kebijakan yang dikelola dan kebijakan inline dalam Panduan Pengguna IAM.

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACL)

Daftar kontrol akses (ACL) mengendalikan prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, lihat Gambaran umum daftar kontrol akses (ACL) dalam Panduan Developer Amazon Simple Storage Service.

Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Jenis-jenis kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh jenis kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan tempat Anda mengatur izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke entitas IAM (pengguna IAM atau peran IAM). Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh batasan izin. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya tentang batasan izin, lihat Batasan izin untuk entitas IAM dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCP) — SCP adalah kebijakan JSON yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU) di. AWS Organizations AWS Organizations adalah layanan untuk mengelompokkan dan mengelola secara terpusat beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur di organisasi, Anda dapat menerapkan kebijakan kontrol layanan (SCP) ke salah satu atau semua akun Anda. SCP membatasi izin untuk entitas di akun anggota, termasuk masing-masing. Pengguna root akun AWS Untuk informasi selengkapnya tentang Organisasi dan SCP, lihat Cara kerja SCP dalam Panduan Pengguna AWS Organizations .
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara programatis untuk peran atau pengguna terfederasi. Izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya, lihat Kebijakan sesi dalam Panduan Pengguna IAM.

Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat Logika evaluasi kebijakan di Panduan Pengguna IAM.

Ikhtisar mengelola akses di Amazon SQS

Setiap AWS sumber daya dimiliki oleh Akun AWS, dan izin untuk membuat atau mengakses sumber daya diatur oleh kebijakan izin. Administrator akun dapat melampirkan kebijakan izin ke identitas

IAM (pengguna, grup, dan peran), dan beberapa layanan (seperti Amazon SQS) juga mendukung melampirkan kebijakan izin ke sumber daya.

> **Note**
>
> Administrator akun (atau pengguna administrator) adalah pengguna dengan hak administratif. Untuk informasi selengkapnya, lihat Praktik Terbaik IAM dalam Panduan Pengguna IAM.

Saat memberikan izin, Anda menentukan pengguna apa yang mendapatkan izin, sumber daya yang mereka dapatkan izin, dan tindakan spesifik yang ingin Anda izinkan pada sumber daya.

Topik

- Sumber daya dan operasi Amazon Simple Queue Service
- Memahami kepemilikan sumber daya
- Mengelola akses ke sumber daya
- Menentukan elemen kebijakan: Tindakan, efek, sumber daya, dan prinsipal

Sumber daya dan operasi Amazon Simple Queue Service

Di Amazon SQS, satu-satunya sumber daya adalah antrian. Dalam kebijakan, gunakan Nama Sumber Daya Amazon (ARN) untuk mengidentifikasi sumber daya yang berlaku untuk kebijakan tersebut. Sumber daya berikut memiliki ARN unik yang terkait dengannya:

Jenis sumber daya	Format ARN
Antrean	`arn:aws:sqs:` *`region`*`:`*`account_id`*`:`*`queue_name`*

Berikut ini adalah contoh format ARN untuk antrian:

- ARN untuk antrian yang disebutkan `my_queue` di wilayah AS Timur (Ohio), milik Akun 123456789012: AWS

```
arn:aws:sqs:us-east-2:123456789012:my_queue
```

- ARN untuk antrian yang diberi nama `my_queue` di setiap wilayah berbeda yang didukung Amazon SQS:

```
arn:aws:sqs:*:123456789012:my_queue
```

- ARN yang menggunakan `*` atau `?` sebagai wildcard untuk nama antrian. Dalam contoh berikut, ARN mencocokkan semua antrian yang diawali dengan: `my_prefix_`

```
arn:aws:sqs:*:123456789012:my_prefix_*
```

Anda bisa mendapatkan nilai ARN untuk antrian yang ada dengan memanggil tindakan. `GetQueueAttributes` Nilai `QueueArn` atribut adalah ARN dari antrian. Untuk informasi selengkapnya tentang ARN, lihat ARN IAM di Panduan Pengguna IAM.

Amazon SQS menyediakan serangkaian tindakan yang bekerja dengan sumber daya antrian. Untuk informasi selengkapnya, lihat Izin Amazon SQS API: Tindakan dan referensi sumber daya.

Memahami kepemilikan sumber daya

Akun AWS Memiliki sumber daya yang dibuat di akun, terlepas dari siapa yang menciptakan sumber daya. Secara khusus, pemilik sumber daya adalah entitas utama (yaitu, akun root, pengguna, atau peran IAM) yang mengautentikasi permintaan pembuatan sumber daya. Akun AWS Contoh berikut menggambarkan cara kerjanya:

- Jika Anda menggunakan kredensi akun root Anda Akun AWS untuk membuat antrean Amazon SQS, Akun AWS Anda adalah pemilik sumber daya (di Amazon SQS, sumber dayanya adalah antrian Amazon SQS).
- Jika Anda membuat pengguna di dalam Akun AWS dan memberikan izin untuk membuat antrian ke pengguna, pengguna dapat membuat antrean. Namun, Anda Akun AWS (milik pengguna) memiliki sumber daya antrian.
- Jika Anda membuat peran IAM Akun AWS dengan izin untuk membuat antrean Amazon SQS, siapa pun yang dapat mengambil peran tersebut dapat membuat antrean. Anda Akun AWS (yang menjadi milik peran tersebut) memiliki sumber daya antrian.

Mengelola akses ke sumber daya

Kebijakan izin menjelaskan izin yang diberikan ke akun. Bagian berikut menjelaskan opsi yang tersedia untuk membuat kebijakan izin.

> **Note**
>
> Bagian ini membahas penggunaan IAM dalam konteks Amazon SQS. Bagian ini tidak memberikan informasi yang mendetail tentang layanan IAM. Untuk dokumentasi lengkap IAM, lihat Apa yang Dimaksud dengan IAM? dalam Panduan Pengguna IAM. Untuk informasi tentang sintaksis dan penjelasan kebijakan IAM, lihat Referensi Kebijakan IAM AWS di Panduan Pengguna IAM.

Kebijakan yang terlampir pada identitas IAM disebut kebijakan (kebijakan IAM) berbasis identitas dan kebijakan yang dilampirkan pada sumber daya disebut kebijakan berbasis sumber daya.

Kebijakan berbasis identitas

Ada dua cara untuk memberikan izin kepada pengguna Anda ke antrian Amazon SQS Anda: menggunakan sistem kebijakan Amazon SQS dan menggunakan sistem kebijakan IAM. Anda dapat menggunakan salah satu sistem, atau keduanya, untuk melampirkan kebijakan ke pengguna atau peran. Dalam kebanyakan kasus, Anda dapat mencapai hasil yang sama menggunakan salah satu sistem. Misalnya, Anda dapat melakukan hal berikut:

- Lampirkan kebijakan izin ke pengguna atau grup di akun Anda — Untuk memberikan izin pengguna untuk membuat antrean Amazon SQS, lampirkan kebijakan izin ke pengguna atau grup tempat pengguna tersebut berada.
- Lampirkan kebijakan izin ke pengguna di pengguna lain Akun AWS — Untuk memberikan izin pengguna untuk membuat antrean Amazon SQS, lampirkan kebijakan izin Amazon SQS ke pengguna lain. Akun AWS

  Izin lintas akun tidak berlaku untuk tindakan berikut:

  - `AddPermission`
  - `CancelMessageMoveTask`
  - `CreateQueue`
  - `DeleteQueue`

- ListMessageMoveTask
- ListQueues
- ListQueueTags
- RemovePermission
- SetQueueAttributes
- StartMessageMoveTask
- TagQueue
- UntagQueue

- Lampirkan kebijakan izin ke peran (berikan izin lintas akun) — Untuk memberikan izin lintas akun, lampirkan kebijakan izin berbasis identitas ke peran IAM. Misalnya, administrator Akun AWS A dapat membuat peran untuk memberikan izin lintas akun ke Akun AWS B (atau AWS layanan) sebagai berikut:
  - Akun Administrator membuat peran IAM dan melampirkan kebijakan izin — yang memberikan izin pada sumber daya di akun A — ke peran tersebut.
  - Administrator akun A melampirkan kebijakan kepercayaan ke peran yang mengidentifikasi akun B sebagai kepala sekolah yang dapat mengambil peran tersebut.
  - Administrator akun B mendelegasikan izin untuk mengambil peran kepada setiap pengguna di akun B. Hal ini memungkinkan pengguna di akun B untuk membuat atau mengakses antrian di akun A.

    > **Note**
    >
    > Jika Anda ingin memberikan izin untuk mengambil peran ke AWS layanan, kepala sekolah dalam kebijakan kepercayaan juga dapat menjadi kepala AWS layanan.

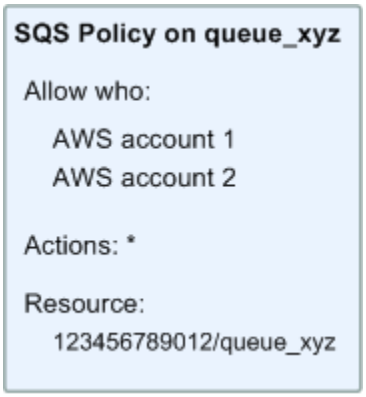
Untuk informasi selengkapnya tentang penggunaan IAM untuk mendelegasikan izin, lihat Manajemen Akses dalam Panduan Pengguna IAM.

Meskipun Amazon SQS bekerja dengan kebijakan IAM, Amazon SQS memiliki infrastruktur kebijakannya sendiri. Anda dapat menggunakan kebijakan Amazon SQS dengan antrean untuk menentukan AWS Akun mana yang memiliki akses ke antrean. Anda dapat menentukan jenis akses dan kondisi (misalnya, kondisi yang memberikan izin untuk digunakan`SendMessage`, `ReceiveMessage` jika permintaan dibuat sebelum 31 Desember 2010). Tindakan spesifik yang dapat Anda berikan izin adalah bagian dari daftar keseluruhan tindakan Amazon SQS. Saat Anda

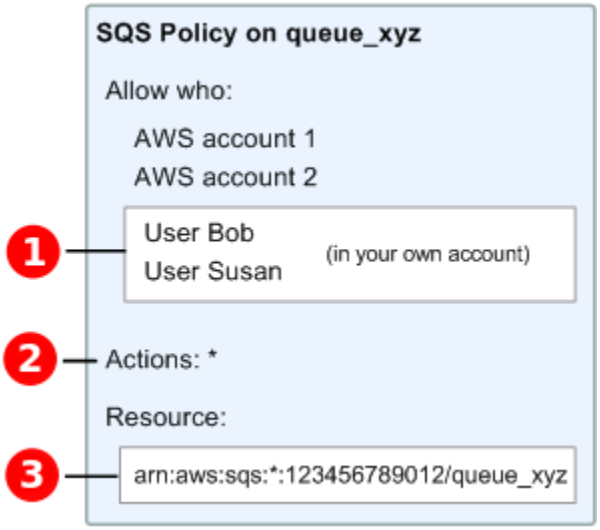
menulis kebijakan Amazon SQS dan menentukan * “izinkan semua tindakan Amazon SQS,” itu berarti pengguna dapat melakukan semua tindakan dalam subset ini.

Diagram berikut mengilustrasikan konsep salah satu kebijakan Amazon SQS dasar ini yang mencakup subset tindakan. Kebijakan ini untuk`queue_xyz`, dan memberikan izin AWS Akun 1 dan AWS Akun 2 untuk menggunakan tindakan apa pun yang diizinkan dengan antrian yang ditentukan.

> **Note**
>
> Sumber daya dalam kebijakan ditentukan sebagai`123456789012/queue_xyz`, di `123456789012` mana ID AWS Akun akun yang memiliki antrian.



Dengan diperkenalkannya IAM dan konsep Pengguna dan Nama Sumber Daya Amazon (ARN), beberapa hal telah berubah tentang kebijakan SQS. Diagram dan tabel berikut menjelaskan perubahannya.



1 Untuk informasi tentang memberikan izin kepada pengguna di akun yang berbeda, lihat Tutorial: Mendelegasikan Akses di Seluruh AWS Akun Menggunakan Peran IAM di Panduan Pengguna IAM.

2 Subset tindakan yang termasuk dalam * telah diperluas. Untuk daftar tindakan yang diizinkan, lihatIzin Amazon SQS API: Tindakan dan referensi sumber daya.

3 Anda dapat menentukan sumber daya menggunakan Amazon Resource Name (ARN), sarana standar untuk menentukan sumber daya dalam kebijakan IAM. Untuk informasi tentang format ARN untuk antrian Amazon SQS, lihat. Sumber daya dan operasi Amazon Simple Queue Service

Misalnya, menurut kebijakan Amazon SQS pada diagram sebelumnya, siapa pun yang memiliki kredensi keamanan untuk AWS Akun 1 atau Akun 2 dapat mengakses. AWS `queue_xyz` Selain itu, Pengguna Bob dan Susan di AWS Akun Anda sendiri (dengan ID`123456789012`) dapat mengakses antrian.

Sebelum pengenalan IAM, Amazon SQS secara otomatis memberi pembuat antrian kontrol penuh atas antrian (yaitu, akses ke semua kemungkinan tindakan Amazon SQS pada antrian itu). Ini tidak lagi benar, kecuali pembuatnya menggunakan kredensi AWS keamanan. Setiap pengguna yang memiliki izin untuk membuat antrian juga harus memiliki izin untuk menggunakan tindakan Amazon SQS lainnya untuk melakukan apa pun dengan antrian yang dibuat.

Berikut ini adalah contoh kebijakan yang memungkinkan pengguna untuk menggunakan semua tindakan Amazon SQS, tetapi hanya dengan antrian yang namanya diawali dengan string literal. `bob_queue_`

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:*:123456789012:bob_queue_*"
   }]
}
```

Untuk informasi selengkapnya, lihatMenggunakan kebijakan dengan Amazon SQS, dan Identitas (Pengguna, Grup, dan Peran) di Panduan Pengguna IAM.

Menentukan elemen kebijakan: Tindakan, efek, sumber daya, dan prinsipal

Untuk setiap sumber daya Amazon Simple Queue Service, layanan mendefinisikan serangkaian tindakan. Untuk memberikan izin untuk tindakan ini, Amazon SQS mendefinisikan serangkaian tindakan yang dapat Anda tentukan dalam kebijakan.

> **Note**
>
> Melakukan tindakan dapat memerlukan izin untuk lebih dari satu tindakan. Saat memberikan izin untuk tindakan tertentu, Anda juga mengidentifikasi sumber daya tempat tindakan diizinkan atau ditolak.

Berikut adalah elemen-elemen kebijakan yang paling dasar:

- Sumber daya – Dalam kebijakan, Anda menggunakan Amazon Resource Name (ARN) untuk mengidentifikasi sumber daya yang diatur kebijakan.
- Tindakan – Anda menggunakan kata kunci tindakan untuk mengidentifikasi tindakan sumber daya yang ingin Anda izinkan atau tolak. Misalnya, `sqs:CreateQueue` izin memungkinkan pengguna untuk melakukan `CreateQueue` tindakan Amazon Simple Queue Service.
- Efek – Anda menentukan efek ketika pengguna meminta tindakan tertentu—efek ini dapat berupa pemberian izin atau penolakan. Jika Anda tidak secara eksplisit memberikan akses ke sumber daya, akses secara implisit ditolak. Anda juga dapat secara eksplisit menolak akses ke sumber daya, yang mungkin Anda lakukan untuk memastikan bahwa pengguna tidak dapat mengaksesnya, meskipun kebijakan lain memberikan akses.
- Prinsipal – Dalam kebijakan berbasis identitas (Kebijakan IAM), pengguna yang dilampiri kebijakan adalah prinsipal secara implisit. Untuk kebijakan berbasis sumber daya, Anda menentukan pengguna, akun, layanan, atau entitas lain yang diinginkan untuk menerima izin (berlaku hanya untuk kebijakan berbasis sumber daya).

Untuk mempelajari selengkapnya tentang sintaks dan deskripsi kebijakan Amazon SQS, lihat Referensi Kebijakan AWS IAM di Panduan Pengguna IAM.

Untuk tabel semua tindakan Layanan Antrian Sederhana Amazon dan sumber daya yang diterapkan, lihatIzin Amazon SQS API: Tindakan dan referensi sumber daya.

Bagaimana Amazon Simple Queue Service bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke Amazon SQS, pelajari fitur IAM apa yang tersedia untuk digunakan dengan Amazon SQS.

Fitur IAM yang dapat Anda gunakan dengan Amazon Simple Queue Service

Fitur IAM	Dukungan Amazon SQS
Kebijakan berbasis identitas	Ya
Kebijakan berbasis sumber daya	Ya
Tindakan kebijakan	Ya
Sumber daya kebijakan	Ya
kunci-kunci persyaratan kebijakan (spesifik layanan)	Ya
ACL	Tidak
ABAC (tanda dalam kebijakan)	Parsial
Kredensial sementara	Ya
Sesi akses teruskan (FAS)	Ya
Peran layanan	Ya
Peran terkait layanan	Tidak

Untuk mendapatkan tampilan tingkat tinggi tentang cara kerja Amazon SQS dan layanan AWS lainnya dengan sebagian besar fitur IAM, AWS lihat layanan yang bekerja dengan IAM di Panduan Pengguna IAM.

Kontrol akses

Daftar kontrol akses (ACL) mengendalikan prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, lihat Gambaran umum daftar kontrol akses (ACL) dalam Panduan Developer Amazon Simple Storage Service.

> **Note**
>
> Penting untuk dipahami bahwa semua Akun AWS dapat mendelegasikan izin mereka kepada pengguna di bawah akun mereka. Akses lintas akun memungkinkan Anda berbagi akses ke AWS sumber daya Anda tanpa harus mengelola pengguna tambahan. Untuk informasi tentang penggunaan akses lintas akun, lihat Mengaktifkan Akses Lintas Akun dalam Panduan Pengguna IAM.
> Lihat Batasan kebijakan kustom Amazon SQS untuk detail lebih lanjut tentang izin lintas konten dan kunci kondisi dalam kebijakan khusus Amazon SQS.

Kebijakan berbasis identitas untuk Amazon SQS

Mendukung kebijakan berbasis identitas	Ya

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta kondisi yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Anda tidak dapat menentukan secara spesifik prinsipal dalam sebuah kebijakan berbasis identitas karena prinsipal berlaku bagi pengguna atau peran yang melekat kepadanya. Untuk mempelajari semua elemen yang dapat Anda gunakan dalam kebijakan JSON, lihat Referensi elemen kebijakan JSON IAM dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk Amazon SQS

Untuk melihat contoh kebijakan berbasis identitas Amazon SQS, lihat. Praktik terbaik kebijakan

Kebijakan berbasis sumber daya dalam Amazon SQS

Mendukung kebijakan berbasis sumber daya	Ya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan secara spesifik seluruh akun atau entitas IAM di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya. Menambahkan prinsipal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsipal dan sumber daya berbeda Akun AWS, administrator IAM di akun tepercaya juga harus memberikan izin entitas utama (pengguna atau peran) untuk mengakses sumber daya. Mereka memberikan izin dengan melampirkan kebijakan berbasis identitas kepada entitas. Namun, jika kebijakan berbasis sumber daya memberikan akses ke prinsipal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk informasi selengkapnya, lihat Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM.

Tindakan kebijakan untuk Amazon SQS

Mendukung tindakan kebijakan	Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan operasi AWS API terkait. Ada beberapa pengecualian, misalnya tindakan hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Menyertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Untuk melihat daftar tindakan Amazon SQS, lihat Sumber Daya yang Ditentukan oleh Amazon Simple Queue Service di Referensi Otorisasi Layanan.

Tindakan kebijakan di Amazon SQS menggunakan awalan berikut sebelum tindakan:

```
sqs
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan tersebut dengan koma.

```
"Action": [
      "sqs:action1",
      "sqs:action2"
         ]
```

Untuk melihat contoh kebijakan berbasis identitas Amazon SQS, lihat. Praktik terbaik kebijakan

Sumber daya kebijakan untuk Amazon SQS

Mendukung sumber daya kebijakan	Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek yang menjadi target penerapan tindakan. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`. Praktik terbaiknya, tentukan sumber daya menggunakan Amazon Resource Name (ARN). Anda dapat melakukan ini untuk

tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": "*"
```

Untuk melihat daftar jenis sumber daya Amazon SQS dan ARNnya, lihat Tindakan yang Ditentukan oleh Layanan Antrian Sederhana Amazon di Referensi Otorisasi Layanan. Untuk mempelajari tindakan yang dapat Anda tentukan ARN dari setiap sumber daya, lihat Sumber Daya yang Ditentukan oleh Amazon Simple Queue Service.

Untuk melihat contoh kebijakan berbasis identitas Amazon SQS, lihat. Praktik terbaik kebijakan

Kunci kondisi kebijakan untuk Amazon SQS

Mendukung kunci kondisi kebijakan khusus layanan	Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan operator kondisi, misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam sebuah pernyataan, atau beberapa kunci dalam elemen `Condition` tunggal, maka AWS akan mengevaluasinya menggunakan operasi `AND` logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan `OR` operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Sebagai contoh, Anda dapat memberikan izin kepada pengguna IAM untuk mengakses sumber daya hanya jika

izin tersebut mempunyai tag yang sesuai dengan nama pengguna IAM mereka. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM: variabel dan tag dalam Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi AWS global, lihat kunci konteks kondisi AWS global di Panduan Pengguna IAM.

Untuk melihat daftar kunci kondisi Amazon SQS, lihat Kunci Kondisi untuk Layanan Antrian Sederhana Amazon di Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat Sumber Daya yang Ditentukan oleh Amazon Simple Queue Service.

Untuk melihat contoh kebijakan berbasis identitas Amazon SQS, lihat. Praktik terbaik kebijakan

ACL di Amazon SQS

Mendukung ACL	Tidak

Daftar kontrol akses (ACL) mengendalikan pengguna utama mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

ABAC dengan Amazon SQS

Mendukung ABAC (tanda dalam kebijakan)	Parsial

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke entitas IAM (pengguna atau peran) dan ke banyak AWS sumber daya. Penandaan ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi ketika tag milik prinsipal cocok dengan tag yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna di situasi saat manajemen kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tag, berikan informasi tentang tag di elemen kondisi dari kebijakan menggunakan kunci kondisi `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, atau `aws:TagKeys`.

Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya tentang ABAC, lihat Apa itu ABAC? dalam Panduan Pengguna IAM. Untuk melihat tutorial yang menguraikan langkah-langkah pengaturan ABAC, lihat Menggunakan kontrol akses berbasis atribut (ABAC) dalam Panduan Pengguna IAM.

Menggunakan kredensi sementara dengan Amazon SQS

Mendukung penggunaan kredensial sementara	Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensi sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensi sementara, lihat Layanan AWS yang bekerja dengan IAM di Panduan Pengguna IAM.

Anda menggunakan kredensi sementara jika Anda masuk AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS menggunakan tautan masuk tunggal (SSO) perusahaan Anda, proses tersebut secara otomatis membuat kredensi sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna lalu beralih peran. Untuk informasi selengkapnya tentang peralihan peran, lihat Peralihan peran (konsol) dalam Panduan Pengguna IAM.

Anda dapat membuat kredensyal sementara secara manual menggunakan API AWS CLI atau AWS . Anda kemudian dapat menggunakan kredensi sementara tersebut untuk mengakses. AWS AWS merekomendasikan agar Anda secara dinamis menghasilkan kredensi sementara alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat Kredensial keamanan sementara di IAM.

Teruskan sesi akses untuk Amazon SQS

Mendukung sesi akses maju (FAS)	Ya

Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah

tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.

Peran layanan untuk Amazon SQS

Mendukung peran layanan	Ya

Peran layanan adalah peran IAM yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Membuat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

> **Warning**
>
> Mengubah izin untuk peran layanan dapat merusak fungsionalitas Amazon SQS. Edit peran layanan hanya jika Amazon SQS memberikan panduan untuk melakukannya.

Peran terkait layanan untuk Amazon SQS

Mendukung peran terkait layanan	Tidak

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Untuk detail tentang pembuatan atau manajemen peran terkait layanan, lihat Layanan AWS yang berfungsi dengan IAM. Cari layanan dalam tabel yang memiliki `Yes` di kolom Peran terkait layanan. Pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Pembaruan Amazon SQS ke AWS kebijakan terkelola

Menambahkan izin ke para pengguna, grup, dan peran lebih mudah dilakukan dengan menggunakan kebijakan terkelola AWS dibandingkan dengan menulis kebijakan sendiri. Dibutuhkan waktu dan keahlian untuk membuat kebijakan terkelola pelanggan IAM yang hanya menyediakan izin sesuai kebutuhan tim Anda. Untuk mulai dengan cepat, Anda dapat menggunakan kebijakan-kebijakan terkelola AWS kami. Kebijakan-kebijakan ini mencakup kasus penggunaan umum dan tersedia di akun AWS Anda. Untuk informasi selengkapnya tentang kebijakan AWS AWS terkelola, lihat kebijakan terkelola di Panduan Pengguna IAM.

AWS layanan memelihara dan memperbarui kebijakan AWS terkelola. Anda tidak dapat mengubah izin dalam kebijakan AWS terkelola. Layanan terkadang menambahkan izin tambahan ke kebijakan AWS terkelola untuk mendukung fitur baru. Jenis pembaruan ini akan memengaruhi semua identitas (pengguna, grup, dan peran) di mana kebijakan tersebut dilampirkan. Layanan kemungkinan besar akan memperbarui kebijakan AWS terkelola saat fitur baru diluncurkan atau saat operasi baru tersedia. Layanan tidak menghapus izin dari kebijakan AWS terkelola, sehingga pembaruan kebijakan tidak akan merusak izin yang ada.

Selain itu, AWS mendukung kebijakan terkelola untuk fungsi pekerjaan yang mencakup beberapa layanan. Misalnya, kebijakan AWS terkelola ReadOnlyAccess menyediakan akses hanya-baca ke semua AWS layanan dan sumber daya. Saat layanan meluncurkan fitur baru, AWS tambahkan izin hanya-baca untuk operasi dan sumber daya baru. Untuk melihat daftar dan deskripsi dari kebijakan fungsi tugas, lihat kebijakan yang dikelola AWS untuk fungsi tugas di Panduan Pengguna IAM.

AWS kebijakan terkelola: AmazonSQS FullAccess

Anda dapat melampirkan `AmazonSQSFullAccess` kebijakan ke identitas Amazon SQS Anda. Kebijakan ini memberikan izin yang memungkinkan akses penuh ke Amazon SQS.

Untuk melihat izin kebijakan ini, lihat AmazonSQS FullAccess di Referensi Kebijakan Terkelola.AWS

AWS kebijakan terkelola: Akses AmazonSQS ReadOnly

Anda dapat melampirkan `AmazonSQSReadOnlyAccess` kebijakan ke identitas Amazon SQS Anda. Kebijakan ini memberikan izin yang memungkinkan akses hanya-baca ke Amazon SQS.

Untuk melihat izin kebijakan ini, lihat ReadOnlyAkses AmazonSQS di Referensi Kebijakan Terkelola.AWS

Pembaruan Amazon SQS ke AWS kebijakan terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk Amazon SQS sejak layanan ini mulai melacak perubahan ini. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan umpan RSS di halaman riwayat Dokumen Amazon SQS.

Perubahan	Deskripsi	Tanggal
Akses AmazonSQS ReadOnly	Amazon SQS menambahkan tindakan baru yang memungkinkan Anda mencantumkan tugas pergerakan pesan terbaru (hingga 10) di bawah antrian sumber tertentu. Tindakan ini dikaitkan dengan operasi API `ListMessageMoveTasks` .	9 Juni 2023

Memecahkan masalah identitas dan akses Amazon Simple Queue Service

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan Amazon SQS dan IAM.

Topik

- Saya tidak berwenang untuk melakukan tindakan di Amazon SQS
- Saya tidak berwenang untuk melakukan iam: PassRole
- Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses sumber daya Amazon SQS saya

Saya tidak berwenang untuk melakukan tindakan di Amazon SQS

Jika Anda menerima pesan kesalahan bahwa Anda tidak memiliki otorisasi untuk melakukan tindakan, kebijakan Anda harus diperbarui agar Anda dapat melakukan tindakan tersebut.

Contoh kesalahan berikut terjadi ketika pengguna `mateojackson` mencoba menggunakan konsol untuk melihat detail tentang suatu sumber daya *`my-example-widget`* fiktif, tetapi tidak memiliki izin `sqs:`*`GetWidget`* fiktif.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 sqs:GetWidget on resource: my-example-widget
```

Dalam hal ini, kebijakan Mateo harus diperbarui untuk memungkinkannya mengakses *`my-example-widget`* sumber daya menggunakan `sqs:`*`GetWidget`* tindakan.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya tidak berwenang untuk melakukan iam: PassRole

Jika Anda menerima kesalahan yang tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran ke Amazon SQS.

Beberapa Layanan AWS memungkinkan Anda untuk meneruskan peran yang ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait layanan. Untuk melakukannya, Anda harus memiliki izin untuk meneruskan peran ke layanan.

Contoh kesalahan berikut terjadi ketika pengguna IAM bernama `marymajor` mencoba menggunakan konsol untuk melakukan tindakan di Amazon SQS. Namun, tindakan tersebut memerlukan layanan untuk mendapatkan izin yang diberikan oleh peran layanan. Mary tidak memiliki izin untuk meneruskan peran tersebut pada layanan.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dalam kasus ini, kebijakan Mary harus diperbarui agar dia mendapatkan izin untuk melakukan tindakan `iam:PassRole` tersebut.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses sumber daya Amazon SQS saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang

dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACL), Anda dapat menggunakan kebijakan tersebut untuk memberi orang akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa referensi berikut:

- Untuk mengetahui apakah Amazon SQS mendukung fitur-fitur ini, lihat. Bagaimana Amazon Simple Queue Service bekerja dengan IAM
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda di seluruh sumber daya Akun AWS yang Anda miliki, lihat Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki di Panduan Pengguna IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda kepada pihak ketiga Akun AWS, lihat Menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat Menyediakan akses ke pengguna terautentikasi eksternal (federasi identitas) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara menggunakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM.

Menggunakan kebijakan dengan Amazon SQS

Topik ini memberikan contoh kebijakan berbasis identitas di mana administrator akun dapat melampirkan kebijakan izin ke identitas IAM (pengguna, grup, dan peran).

> **Important**
>
> Kami menyarankan Anda terlebih dahulu meninjau topik pengantar yang menjelaskan konsep dasar dan opsi yang tersedia bagi Anda untuk mengelola akses ke sumber daya Layanan Antrian Sederhana Amazon Anda. Untuk informasi selengkapnya, lihat Ikhtisar mengelola akses di Amazon SQS.
>
> Dengan pengecualian`ListQueues`, semua tindakan Amazon SQS mendukung izin tingkat sumber daya. Untuk informasi selengkapnya, lihat Izin Amazon SQS API: Tindakan dan referensi sumber daya.

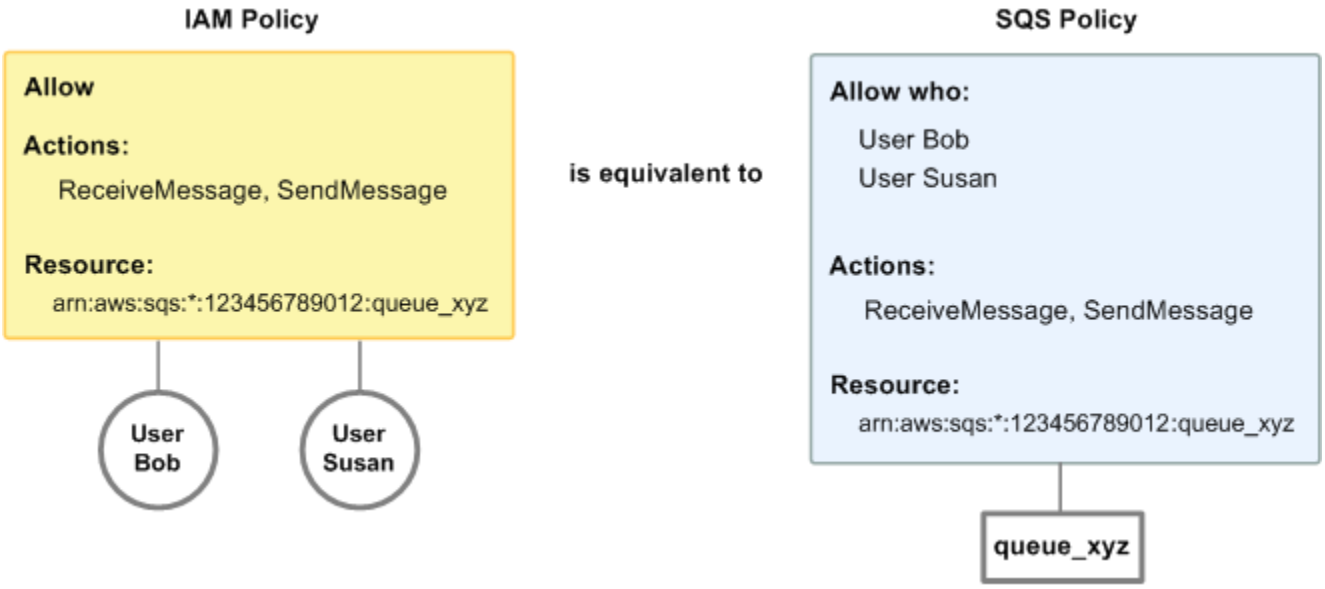
Topik

Menggunakan kebijakan Amazon SQS dan IAM

Ada dua cara untuk memberikan izin kepada pengguna Anda ke sumber daya Amazon SQS Anda: menggunakan sistem kebijakan Amazon SQS dan menggunakan sistem kebijakan IAM. Anda dapat menggunakan salah satu atau yang lain, atau keduanya. Untuk sebagian besar, Anda dapat mencapai hasil yang sama dengan salah satunya.

Misalnya, diagram berikut menunjukkan kebijakan IAM dan kebijakan Amazon SQS yang setara dengannya. Kebijakan IAM memberikan hak atas Amazon `ReceiveMessage` SQS `SendMessage` dan tindakan untuk antrian yang `queue_xyz` dipanggil di Akun AWS Anda, dan kebijakan tersebut dilampirkan ke pengguna bernama Bob dan Susan (Bob dan Susan memiliki izin yang tercantum dalam kebijakan). Kebijakan Amazon SQS ini juga memberi Bob dan Susan hak atas `ReceiveMessage` dan `SendMessage` tindakan untuk antrian yang sama.

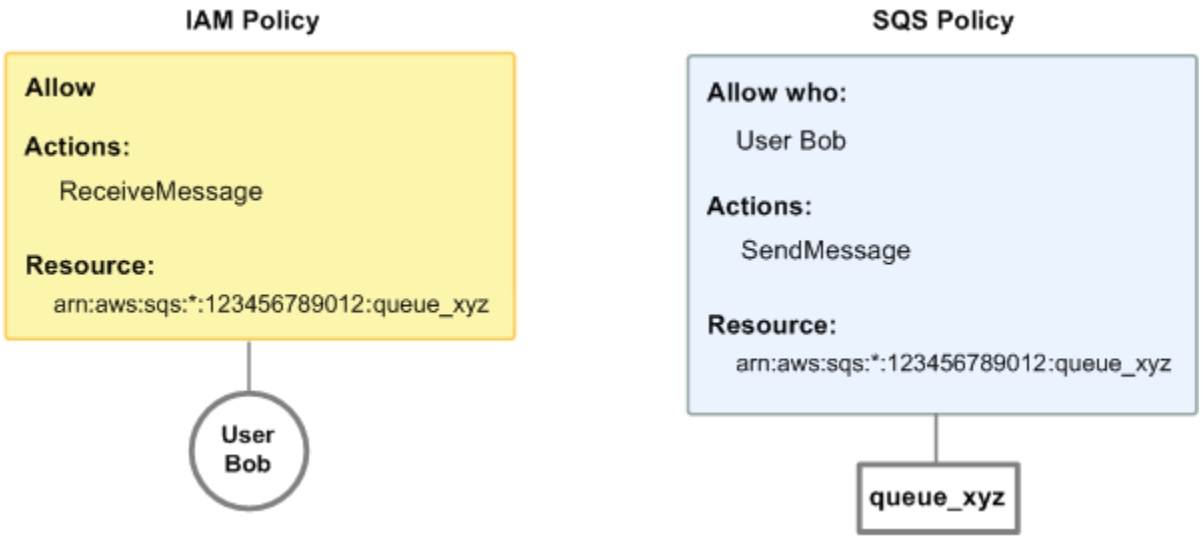
> **Note**
>
> Contoh berikut menunjukkan kebijakan sederhana tanpa kondisi. Anda dapat menentukan kondisi tertentu di salah satu kebijakan dan mendapatkan hasil yang sama.



Ada satu perbedaan utama antara kebijakan IAM dan Amazon SQS: sistem kebijakan Amazon SQS memungkinkan Anda memberikan izin ke Akun AWS lain, sedangkan IAM tidak.

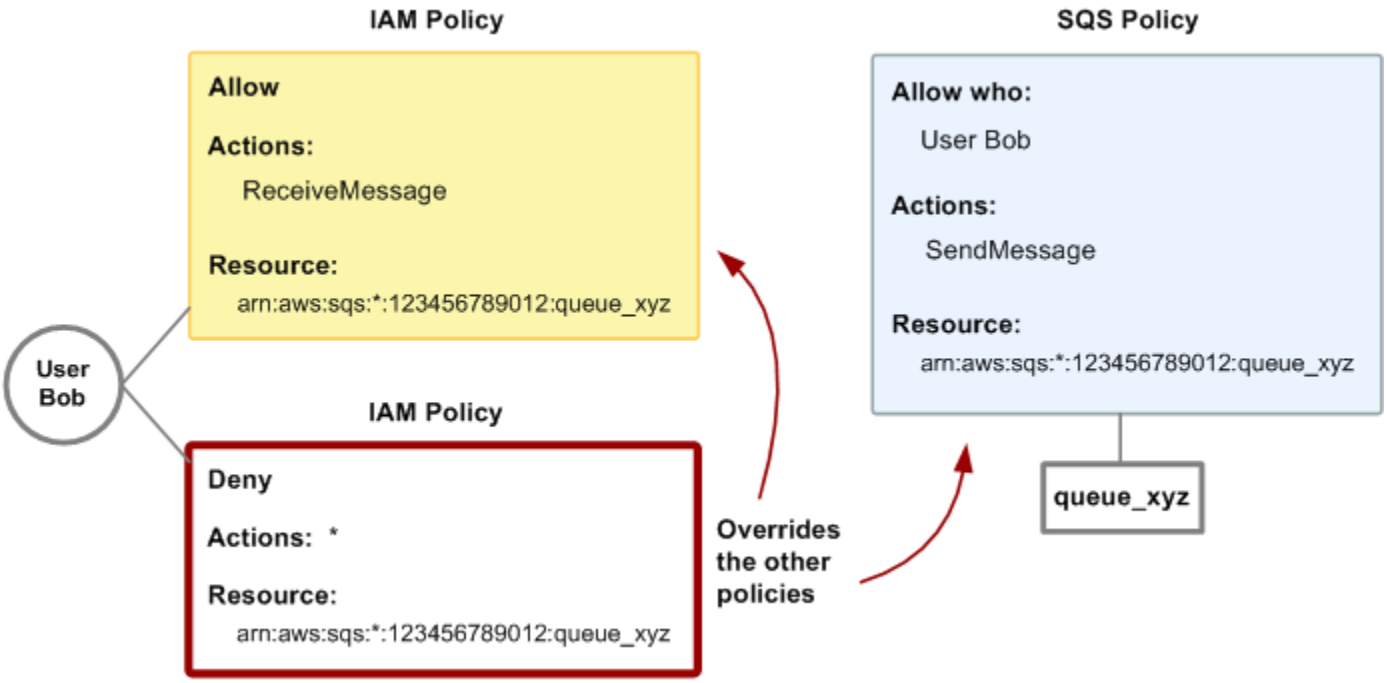
Terserah Anda bagaimana Anda menggunakan kedua sistem bersama-sama untuk mengelola izin Anda. Contoh berikut menunjukkan cara sistem dua kebijakan bekerja sama.

- Dalam contoh pertama, Bob memiliki kebijakan IAM dan kebijakan Amazon SQS yang berlaku untuk akunnya. Kebijakan IAM memberikan izin akunnya untuk `ReceiveMessage` tindakan tersebut`queue_xyz`, sedangkan kebijakan Amazon SQS memberikan izin akunnya untuk tindakan pada antrean `SendMessage` yang sama. Diagram berikut menggambarkan konsep.



Jika Bob mengirim `ReceiveMessage` permintaan kequeue_xyz, kebijakan IAM mengizinkan tindakan tersebut. Jika Bob mengirimkan `SendMessage` permintaan kequeue_xyz, kebijakan Amazon SQS mengizinkan tindakan tersebut.

- Dalam contoh kedua, Bob menyalahgunakan aksesnyaqueue_xyz, sehingga menjadi perlu untuk menghapus seluruh aksesnya ke antrian. Hal termudah untuk dilakukan adalah menambahkan kebijakan yang menolaknya mengakses semua tindakan untuk antrian. Kebijakan ini mengesampingkan dua lainnya karena eksplisit `deny` selalu mengesampingkan. `allow` Untuk informasi selengkapnya tentang logika evaluasi kebijakan, lihatMenggunakan kebijakan khusus dengan Bahasa Kebijakan Akses Amazon SQS. Diagram berikut menggambarkan konsep.



Anda juga dapat menambahkan pernyataan tambahan ke kebijakan Amazon SQS yang menolak Bob semua jenis akses ke antrian. Ini memiliki efek yang sama dengan menambahkan kebijakan IAM yang menolak akses Bob ke antrian. Untuk contoh kebijakan yang mencakup tindakan dan sumber daya Amazon SQS, lihat. Contoh dasar kebijakan Amazon SQS Untuk informasi selengkapnya tentang menulis kebijakan Amazon SQS, lihat. Menggunakan kebijakan khusus dengan Bahasa Kebijakan Akses Amazon SQS

Izin diperlukan untuk menggunakan konsol Amazon SQS

Pengguna yang ingin bekerja dengan konsol Amazon SQS harus memiliki set izin minimum untuk bekerja dengan antrian Amazon SQS di pengguna. Akun AWS Misalnya, pengguna harus memiliki

izin untuk memanggil `ListQueues` tindakan untuk dapat membuat daftar antrian, atau izin untuk memanggil `CreateQueue` tindakan untuk dapat membuat antrian. Selain izin Amazon SQS, untuk berlangganan antrian Amazon SQS ke topik Amazon SNS, konsol juga memerlukan izin untuk tindakan Amazon SNS.

Jika Anda membuat kebijakan IAM yang lebih ketat daripada izin minimum yang diperlukan, konsol mungkin tidak berfungsi sebagaimana dimaksudkan untuk pengguna dengan kebijakan IAM tersebut.

Anda tidak perlu mengizinkan izin konsol minimum untuk pengguna yang hanya melakukan panggilan ke tindakan Amazon SQS AWS CLI atau Amazon.

Contoh kebijakan berbasis identitas untuk Amazon SQS

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau memodifikasi sumber daya Amazon SQS. Mereka juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian akan dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM menggunakan contoh dokumen kebijakan JSON ini, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh Amazon SQS, termasuk format ARN untuk setiap jenis sumber daya, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk Layanan Antrian Sederhana Amazon di Referensi Otorisasi Layanan.

> **Note**
>
> Saat mengonfigurasi kait siklus hidup untuk Auto Scaling Amazon EC2, Anda tidak perlu menulis kebijakan untuk mengirim pesan ke antrean Amazon SQS. Untuk informasi selengkapnya, lihat Kait Siklus Hidup Auto Scaling Amazon EC2 di Panduan Pengguna Amazon EC2.

Topik

- Praktik terbaik kebijakan
- Menggunakan konsol Amazon SQS

- Mengizinkan pengguna melihat izin mereka sendiri
- Izinkan pengguna membuat antrian
- Izinkan pengembang menulis pesan ke antrian bersama
- Izinkan manajer untuk mendapatkan ukuran umum antrian
- Izinkan mitra mengirim pesan ke antrian tertentu

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya Amazon SQS di akun Anda. Tindakan ini membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat Kebijakan yang dikelola AWS atau Kebijakan yang dikelola AWS untuk fungsi tugas dalam Panduan Pengguna IAM.
- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat Kebijakan dan izin dalam IAM dalam Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Kondisi dalam Panduan Pengguna IAM.
- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM.

IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat Validasi kebijakan IAM Access Analyzer dalam Panduan Pengguna IAM.

- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan. Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi akses API yang dilindungi MFA dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat Praktik terbaik keamanan dalam IAM dalam Panduan Pengguna IAM.

Menggunakan konsol Amazon SQS

Untuk mengakses konsol Amazon Simple Queue Service, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang sumber daya Amazon SQS di Anda. Akun AWS Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna atau peran) dengan kebijakan tersebut.

Anda tidak perlu mengizinkan izin konsol minimum untuk pengguna yang melakukan panggilan hanya ke AWS CLI atau AWS API. Sebagai gantinya, izinkan akses hanya ke tindakan yang sesuai dengan operasi API yang coba mereka lakukan.

Untuk memastikan bahwa pengguna dan peran masih dapat menggunakan konsol Amazon SQS, lampirkan juga kebijakan terkelola Amazon `AmazonSQSReadOnlyAccess` AWS SQS ke entitas. Untuk informasi selengkapnya, lihat Menambah izin untuk pengguna dalam Panduan Pengguna IAM.

Mengizinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan ini di konsol atau menggunakan API atau secara terprogram. AWS CLI 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": "*"
        }
    ]
}
```

Izinkan pengguna membuat antrian

Dalam contoh berikut, kami membuat kebijakan untuk Bob yang memungkinkannya mengakses semua tindakan Amazon SQS, tetapi hanya dengan antrian yang namanya diawali dengan string literal. `alice_queue_`

Amazon SQS tidak secara otomatis memberikan izin kepada pembuat antrian untuk menggunakan antrean. Oleh karena itu, kami harus secara eksplisit memberikan izin Bob untuk menggunakan semua tindakan Amazon SQS selain tindakan dalam kebijakan IAM. `CreateQueue`

```
{
   "Version": "2012-10-17",
```

```
   "Statement": [{
      "Effect": "Allow",
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:*:123456789012:alice_queue_*"
   }]
}
```

Izinkan pengembang menulis pesan ke antrian bersama

Dalam contoh berikut, kami membuat grup untuk pengembang dan melampirkan kebijakan yang memungkinkan grup menggunakan `SendMessage` tindakan Amazon SQS, tetapi hanya dengan antrian milik yang ditentukan Akun AWS dan diberi nama. `MyCompanyQueue`

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "sqs:SendMessage",
      "Resource": "arn:aws:sqs:*:123456789012:MyCompanyQueue"
   }]
}
```

Anda dapat menggunakan * alih-alih `SendMessage` untuk memberikan tindakan berikut kepada prinsipal pada antrian bersama:`ChangeMessageVisibility`,,`DeleteMessage`,`GetQueueAttributes`, `GetQueueUrlReceiveMessage`, dan`SendMessage`.

> **Note**
>
> Meskipun * menyertakan akses yang disediakan oleh jenis izin lain, Amazon SQS mempertimbangkan izin secara terpisah. Misalnya, dimungkinkan untuk memberikan keduanya * dan `SendMessage` izin kepada pengguna, meskipun * termasuk akses yang disediakan oleh`SendMessage`.
>
> Konsep ini juga berlaku ketika Anda menghapus izin. Jika kepala sekolah hanya memiliki * izin, meminta untuk menghapus `SendMessage` izin tidak meninggalkan kepala sekolah dengan segala sesuatunya kecuali izin. Sebaliknya, permintaan tidak berpengaruh, karena kepala sekolah tidak memiliki `SendMessage` izin eksplisit. Untuk meninggalkan kepala sekolah hanya dengan `ReceiveMessage` izin, pertama-tama tambahkan `ReceiveMessage` izin dan kemudian hapus * izin.

Izinkan manajer untuk mendapatkan ukuran umum antrian

Dalam contoh berikut, kami membuat grup untuk manajer dan melampirkan kebijakan yang memungkinkan grup menggunakan `GetQueueAttributes` tindakan Amazon SQS dengan semua antrian milik akun yang ditentukan. AWS

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

Izinkan mitra mengirim pesan ke antrian tertentu

Anda dapat menyelesaikan tugas ini menggunakan kebijakan Amazon SQS atau kebijakan IAM. Jika pasangan Anda memiliki Akun AWS, mungkin akan lebih mudah untuk menggunakan kebijakan Amazon SQS. Namun, setiap pengguna di perusahaan mitra yang memiliki kredenal AWS keamanan dapat mengirim pesan ke antrian. Jika Anda ingin membatasi akses ke pengguna atau aplikasi tertentu, Anda harus memperlakukan mitra seperti pengguna di perusahaan Anda sendiri dan menggunakan kebijakan IAM alih-alih kebijakan Amazon SQS.

Contoh ini melakukan tindakan berikut:

1. Buat grup yang dipanggil WidgetCo untuk mewakili perusahaan mitra.
2. Buat pengguna untuk pengguna atau aplikasi tertentu di perusahaan mitra yang membutuhkan akses.
3. Tambahkan pengguna ke grup .
4. Lampirkan kebijakan yang memberikan akses grup hanya ke `SendMessage` tindakan hanya untuk antrian bernama`WidgetPartnerQueue`.

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

```
         "Resource": "arn:aws:sqs:*:123456789012:WidgetPartnerQueue"
   }]
}
```

Contoh dasar kebijakan Amazon SQS

Bagian ini menunjukkan contoh kebijakan untuk kasus penggunaan Amazon SQS umum.

Anda dapat menggunakan konsol untuk memverifikasi efek setiap kebijakan saat Anda melampirkan kebijakan kepada pengguna. Awalnya, pengguna tidak memiliki izin dan tidak akan dapat melakukan apa pun di konsol. Saat Anda melampirkan kebijakan ke pengguna, Anda dapat memverifikasi bahwa pengguna dapat melakukan berbagai tindakan di konsol.

> **Note**
>
> Kami menyarankan Anda menggunakan dua jendela browser: satu untuk memberikan izin dan yang lainnya untuk masuk ke AWS Management Console menggunakan kredensi pengguna untuk memverifikasi izin saat Anda memberikannya kepada pengguna.

Contoh 1: Berikan satu izin kepada satu Akun AWS

Contoh kebijakan berikut memberikan Akun AWS nomor `111122223333` `SendMessage` izin untuk antrian yang disebutkan `444455556666/queue1` di wilayah AS Timur (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"
   }]
}
```

Contoh 2: Berikan dua izin ke satu Akun AWS

Contoh kebijakan berikut memberikan Akun AWS nomor `111122223333` baik `SendMessage` dan `ReceiveMessage` izin untuk antrian bernama. `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"
   }]
}
```

Contoh 3: Berikan semua izin ke dua Akun AWS

Contoh kebijakan berikut memberikan dua Akun AWS nomor yang berbeda (111122223333dan444455556666) izin untuk menggunakan semua tindakan yang Amazon SQS mengizinkan akses bersama untuk antrian yang `123456789012/queue1` dinamai di wilayah AS Timur (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"
   }]
}
```

Contoh 4: Berikan izin lintas akun untuk peran dan nama pengguna

Contoh kebijakan berikut memberikan izin `111122223333` lintas akun nomor `role1` dan `username1` di bawah Akun AWS nomor untuk menggunakan semua tindakan yang Amazon SQS mengizinkan akses bersama untuk antrian yang `123456789012/queue1` dinamai di wilayah AS Timur (Ohio).

Izin lintas akun tidak berlaku untuk tindakan berikut:

- `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"
   }]
}
```

Contoh 5: Berikan izin kepada semua pengguna

Contoh kebijakan berikut memberikan `ReceiveMessage` izin kepada semua pengguna (pengguna anonim) untuk antrian bernama. `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"
   }]
}
```

Contoh 6: Berikan izin terbatas waktu untuk semua pengguna

Contoh kebijakan berikut memberikan `ReceiveMessage` izin kepada semua pengguna (pengguna anonim) untuk antrian bernama`111122223333/queue1`, tetapi hanya antara pukul 12:00 siang (siang) dan 15:00 pada tanggal 31 Januari 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"
         }
      }
   }]
}
```

Contoh 7: Berikan semua izin kepada semua pengguna dalam rentang CIDR

Contoh kebijakan berikut memberikan izin kepada semua pengguna (pengguna anonim) untuk menggunakan semua kemungkinan tindakan Amazon SQS yang dapat dibagikan untuk antrian `111122223333/queue1` bernama, tetapi hanya jika permintaan berasal dari `192.0.2.0/24` rentang CIDR.

```
{
   "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"
         }
      }
   }]
}
```

Contoh 8: Izin daftar izin dan daftar blokir untuk pengguna dalam rentang CIDR yang berbeda

Contoh kebijakan berikut memiliki dua pernyataan:

- Pernyataan pertama memberikan semua pengguna (pengguna anonim) dalam rentang `192.0.2.0/24` CIDR (kecuali`192.0.2.188`) izin untuk menggunakan `SendMessage` tindakan untuk antrian bernama /queue1. `111122223333`
- Pernyataan kedua memblokir semua pengguna (pengguna anonim) dalam rentang `12.148.72.0/23` CIDR dari menggunakan antrian.

```
{
   "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"
         }
      }
   }]
}
```

Menggunakan kebijakan khusus dengan Bahasa Kebijakan Akses Amazon SQS

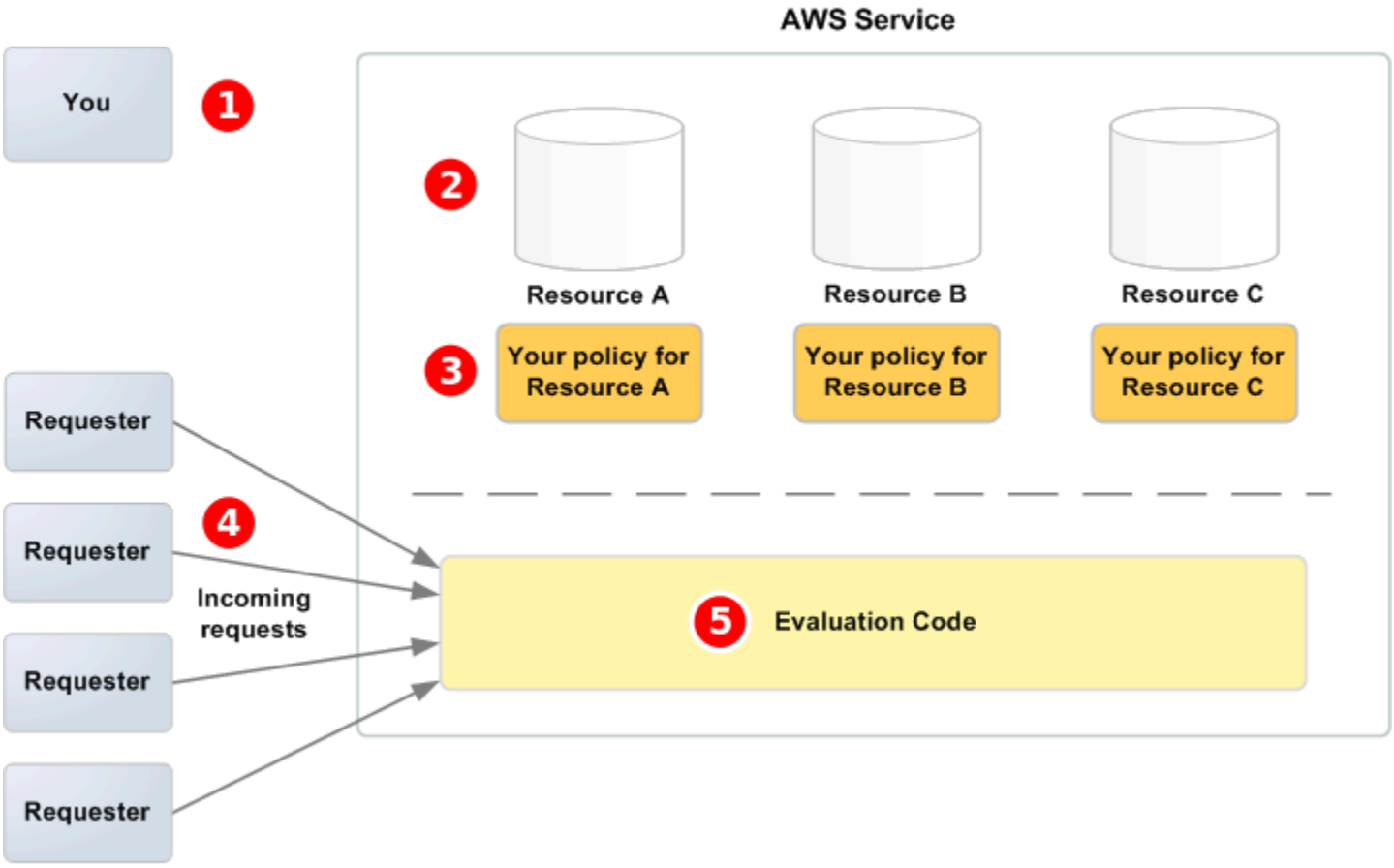
Jika Anda ingin mengizinkan akses Amazon SQS hanya berdasarkan Akun AWS ID dan izin dasar (seperti untuk `SendMessage`atau `ReceiveMessage`), Anda tidak perlu menulis kebijakan Anda sendiri. Anda hanya dapat menggunakan tindakan Amazon SQS. `AddPermission`

Jika Anda ingin secara eksplisit menolak atau mengizinkan akses berdasarkan kondisi yang lebih spesifik (seperti waktu permintaan masuk atau alamat IP pemohon), Anda perlu menulis kebijakan Amazon SQS Anda sendiri dan mengunggahnya ke sistem menggunakan tindakan Amazon SQS. AWS `SetQueueAttributes`

Topik

Arsitektur kontrol akses Amazon SQS

Diagram berikut menjelaskan kontrol akses untuk sumber daya Amazon SQS Anda.



1 Anda, pemilik sumber daya.

2 Sumbe

daya Anda yang terkandung dalam AWS layanan (misalnya, antrian Amazon SQS).

3 Kebijak

Anda. Merupakan praktik yang baik untuk memiliki satu kebijakan per sumber daya. AWS Layanan ini menyediakan API yang Anda gunakan untuk mengunggah dan mengelola kebijakan Anda.

4 Pemoh

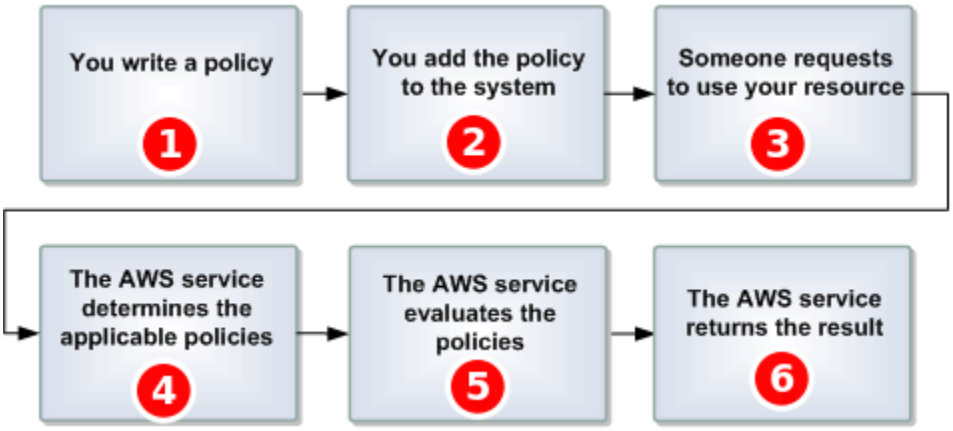
dan permintaan masuk mereka ke layanan. AWS

5 Kode

evaluasi bahasa kebijakan akses. Ini adalah kumpulan kode dalam AWS layanan yang mengevaluasi permintaan masuk terhadap kebijakan yang berlaku dan menentukan apakah pemohon diizinkan mengakses sumber daya.

Alur kerja proses kontrol akses Amazon SQS

Diagram berikut menjelaskan alur kerja umum kontrol akses dengan bahasa kebijakan akses Amazon SQS.



1 Anda

menulis kebijakan Amazon SQS untuk antrian Anda.

2 Anda

mengunggah kebijakan Anda ke AWS. AWS Layanan ini menyediakan API yang Anda gunakan untuk mengunggah kebijakan Anda. Misalnya, Anda menggunakan `SetQueueAttributes` tindakan Amazon SQS untuk mengunggah kebijakan antrian Amazon SQS tertentu.

3 Seseor

mengirim permintaan untuk menggunakan antrian Amazon SQS Anda.

4 Amazo

SQS memeriksa semua kebijakan Amazon SQS yang tersedia dan menentukan kebijakan mana yang berlaku.

5 Amazo

SQS mengevaluasi kebijakan dan menentukan apakah pemohon diizinkan untuk menggunakan antrian Anda.

6 Berdas

hasil evaluasi kebijakan, Amazon SQS mengembalikan `Access denied` kesalahan ke pemohon atau terus memproses permintaan.

Konsep kunci bahasa Kebijakan Akses Amazon SQS

Untuk menulis kebijakan Anda sendiri, Anda harus terbiasa dengan JSON dan sejumlah konsep kunci.

Izinkan

Hasil dari a Pernyataan yang telah Efek diatur `keallow`.

Aksi

Aktivitas yang Kepala Sekolah memiliki izin untuk melakukan, biasanya permintaan untuk AWS.

Default-menyangkal

Hasil dari a Pernyataan yang tidak memiliki Izinkan atau Penyangkalan eksplisit pengaturan.

Kondisi

Pembatasan atau detail apa pun tentang a. Izin Kondisi umum terkait dengan tanggal dan waktu dan alamat IP.

Efek

Hasil yang Anda inginkan Pernyataan dari a Kebijakan untuk kembali pada waktu evaluasi. Anda menentukan `allow` nilai `deny` atau saat Anda menulis pernyataan kebijakan. Ada tiga kemungkinan hasil pada waktu evaluasi kebijakan:Default-menyangkal,Izinkan, danPenyangkalan eksplisit.

Penyangkalan eksplisit

Hasil dari a Pernyataan yang telah Efek diatur kedeny.

Evaluasi

Proses yang digunakan Amazon SQS untuk menentukan apakah permintaan yang masuk harus ditolak atau diizinkan berdasarkan file. Kebijakan

Penerbit

Pengguna yang menulis Kebijakan untuk memberikan izin ke sumber daya. Penerbit, menurut definisi selalu pemilik sumber daya. AWS tidak mengizinkan pengguna Amazon SQS untuk membuat kebijakan untuk sumber daya yang tidak mereka miliki.

Kunci

Karakteristik spesifik yang menjadi dasar pembatasan akses.

Izin

Konsep mengizinkan atau melarang akses ke sumber daya menggunakan a Kondisi dan aKunci.

Kebijakan

Dokumen yang bertindak sebagai wadah untuk satu atau lebih pernyataan.

Amazon SQS menggunakan kebijakan untuk menentukan apakah akan memberikan akses ke pengguna untuk sumber daya.

Kepala Sekolah

Pengguna yang menerima Izin diKebijakan.

Sumber Daya

Objek yang Kepala Sekolah meminta akses ke.

Pernyataan

Deskripsi formal dari izin tunggal, ditulis dalam bahasa kebijakan akses sebagai bagian dari Kebijakan dokumen yang lebih luas.

Pemohon

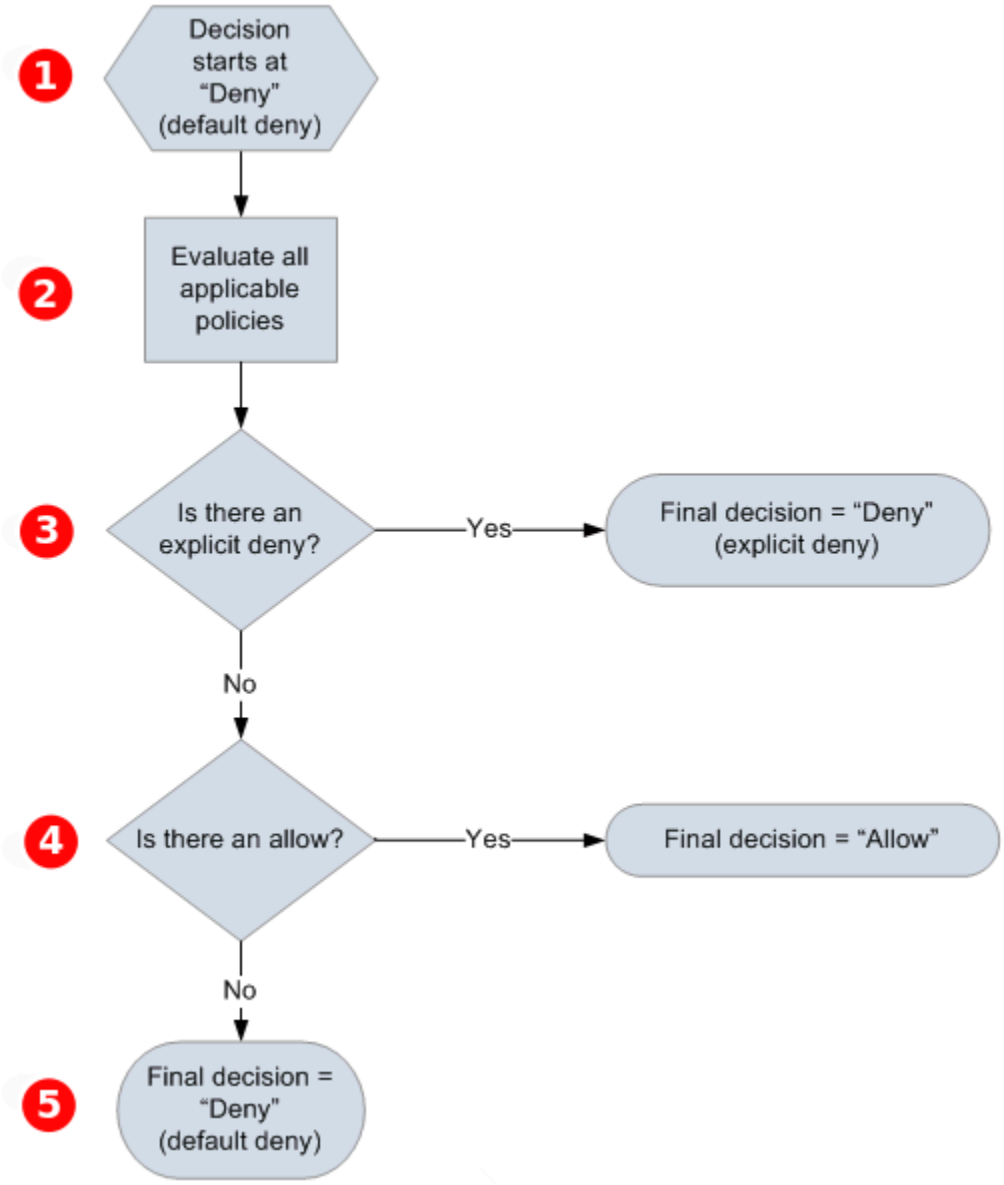
Pengguna yang mengirimkan permintaan akses ke fileSumber Daya.

Logika evaluasi bahasa Kebijakan Akses Amazon SQS

Pada waktu evaluasi, Amazon SQS menentukan apakah permintaan dari orang lain selain pemilik sumber daya harus diizinkan atau ditolak. Logika evaluasi mengikuti beberapa aturan dasar:

- Secara default, semua permintaan untuk menggunakan sumber daya Anda berasal dari siapa pun kecuali Anda ditolak.
- Sebuah Izinkanmengesampingkan apapun. Default-menyangkal
- Sebuah Penyangkalan eksplisitmengesampingkan izin apa pun.
- Urutan di mana kebijakan dievaluasi tidak penting.

Diagram berikut menjelaskan secara rinci bagaimana Amazon SQS mengevaluasi keputusan tentang izin akses.



1 Keputu
dimulai dengan default-deny.

2 Kode
penegakan mengevaluasi semua kebijakan yang berlaku untuk permintaan (berdasarkan sumber daya, prinsip, tindakan, dan kondisi). Urutan kode penegakan mengevaluasi kebijakan tidak penting.

3 Kode
penegakan mencari instruksi penolakan eksplisit yang dapat diterapkan pada permintaan. Jika menemukan bahkan satu, kode penegakan mengembalikan keputusan penolakan dan proses selesai.

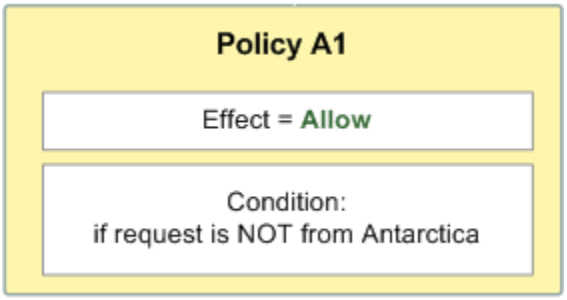
4 Jika tidak ada instruksi penolakan eksplisit yang ditemukan, kode penegakan mencari instruksi izin apa pun yang dapat diterapkan pada permintaan. Jika menemukan satu pun, kode penegakan mengembalikan keputusan izin dan proses selesai (layanan terus memproses permintaan).

5 Jika tidak ada instruksi izinkan ditemukan, maka keputusan akhir ditolak (karena tidak ada penolakan eksplisit atau izinkan, ini dianggap sebagai penolakan default).

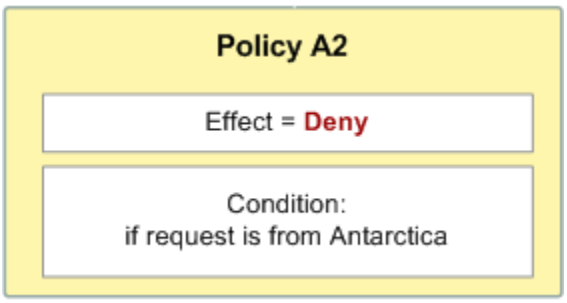
Hubungan antara penolakan eksplisit dan default dalam Bahasa Kebijakan Akses Amazon SQS

Jika kebijakan Amazon SQS tidak langsung berlaku untuk permintaan, permintaan akan menghasilkan. Default-menyangkal Misalnya, jika pengguna meminta izin untuk menggunakan Amazon SQS tetapi satu-satunya kebijakan yang berlaku untuk pengguna dapat menggunakan DynamoDB, permintaan akan menghasilkan default-deny.

Jika kondisi dalam pernyataan tidak terpenuhi, permintaan akan menghasilkan default-deny. Jika semua kondisi dalam pernyataan terpenuhi, permintaan akan menghasilkan suatu Izinkanatau Penyangkalan eksplisitberdasarkan nilai Efekelemen kebijakan. Kebijakan tidak menentukan apa yang harus dilakukan jika kondisi tidak terpenuhi, jadi hasil default dalam kasus ini adalah default-deny. Misalnya, Anda ingin mencegah permintaan yang datang dari Antartika. Anda menulis Kebijakan A1 yang mengizinkan permintaan hanya jika tidak berasal dari Antartika. Diagram berikut menggambarkan kebijakan Amazon SQS.

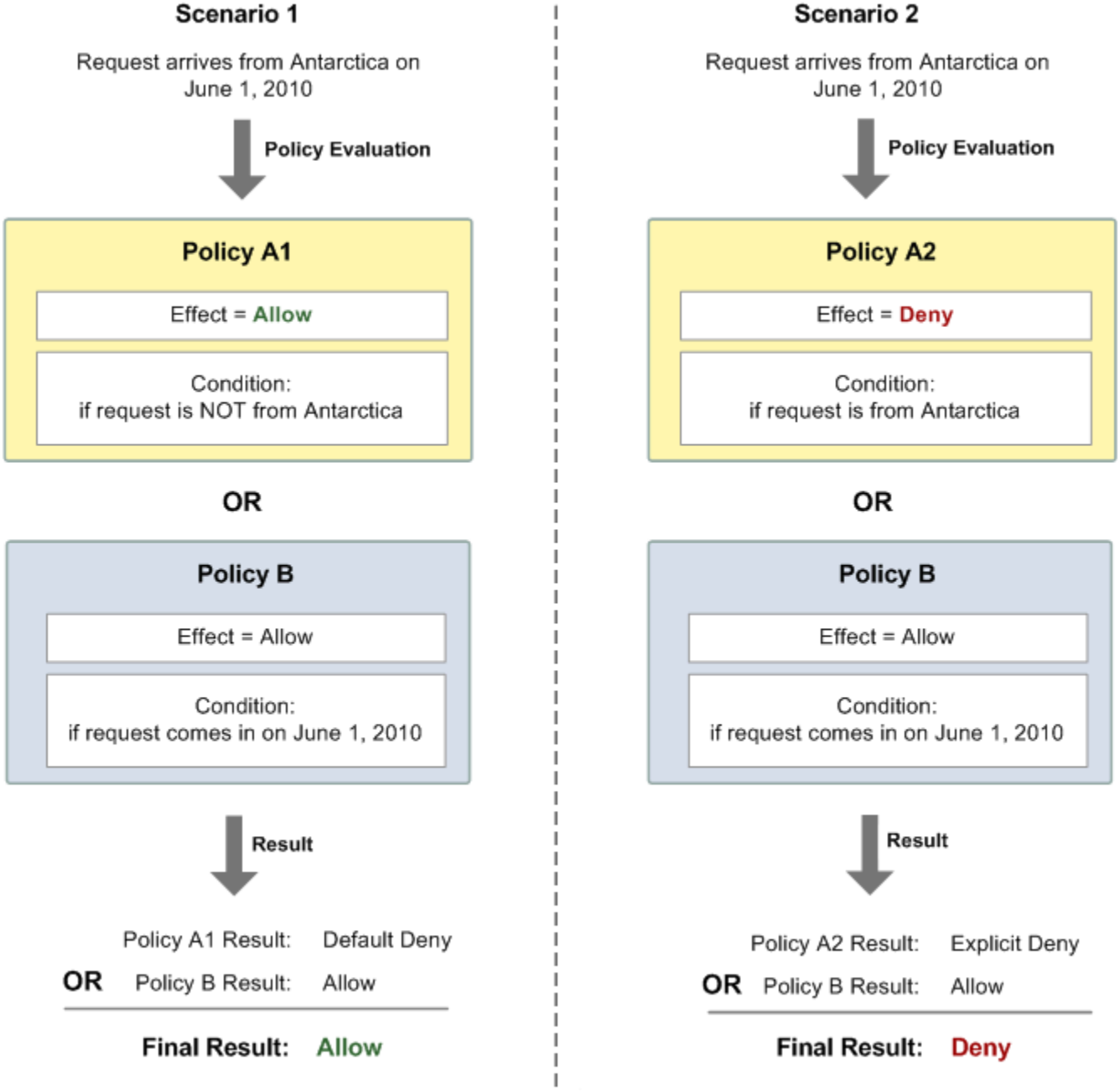


Jika pengguna mengirim permintaan dari AS, kondisi terpenuhi (permintaan bukan dari Antartika), dan permintaan tersebut menghasilkan izin. Namun, jika pengguna mengirim permintaan dari Antartika, kondisi tidak terpenuhi dan permintaan default ke default-deny. Anda dapat mengubah hasilnya menjadi penolakan eksplisit dengan menulis Kebijakan A2 yang secara eksplisit menolak permintaan jika berasal dari Antartika. Diagram berikut menggambarkan kebijakan.



Jika pengguna mengirim permintaan dari Antartika, kondisi terpenuhi dan permintaan tersebut menghasilkan penolakan eksplisit.

Perbedaan antara default-deny dan explicit-deny penting karena allow dapat menimpa yang pertama tetapi bukan yang terakhir. Misalnya, Kebijakan B mengizinkan permintaan jika mereka tiba pada tanggal 1 Juni 2010. Diagram berikut membandingkan penggabungan kebijakan ini dengan Kebijakan A1 dan Kebijakan A2.



Dalam Skenario 1, Kebijakan A1 menghasilkan penolakan default dan Kebijakan B menghasilkan izin karena kebijakan mengizinkan permintaan yang masuk pada tanggal 1 Juni 2010. Izin dari Kebijakan B mengesampingkan default-deny dari Kebijakan A1, dan permintaan diizinkan.

Dalam Skenario 2, Kebijakan B2 menghasilkan penolakan eksplisit dan Kebijakan B menghasilkan izin. Penolakan eksplisit dari Kebijakan A2 mengesampingkan izin dari Kebijakan B, dan permintaan ditolak.

Batasan kebijakan kustom Amazon SQS

Akses lintas akun

Izin lintas akun tidak berlaku untuk tindakan berikut:

- `AddPermission`
- `CancelMessageMoveTask`
- `CreateQueue`
- `DeleteQueue`
- `ListMessageMoveTask`
- `ListQueues`
- `ListQueueTags`
- `RemovePermission`
- `SetQueueAttributes`
- `StartMessageMoveTask`
- `TagQueue`
- `UntagQueue`

Kunci syarat

Saat ini, Amazon SQS hanya mendukung subset terbatas dari kunci kondisi yang tersedia di IAM. Untuk informasi selengkapnya, lihat Izin Amazon SQS API: Tindakan dan referensi sumber daya.

Contoh Bahasa Kebijakan Akses Amazon SQS Kustom

Berikut ini adalah contoh kebijakan akses Amazon SQS yang khas.

Contoh 1: Berikan izin ke satu akun

Contoh berikut kebijakan Amazon SQS memberikan Akun AWS 111122223333 izin untuk mengirim dan menerima dari dimiliki oleh 444455556666. `queue2` Akun AWS

```
{
   "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"
   }]
}
```

Contoh 2: Berikan izin ke satu atau beberapa akun

Contoh berikut kebijakan Amazon SQS memberikan satu atau lebih Akun AWS akses ke antrian yang dimiliki oleh akun Anda untuk jangka waktu tertentu. Penting untuk menulis kebijakan ini dan mengunggahnya ke Amazon SQS menggunakan `SetQueueAttributes`tindakan karena `AddPermission`tindakan tidak mengizinkan penetapan batasan waktu saat memberikan akses ke antrian.

```
{
   "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"
         }
      }
   }]
}
```

Contoh 3: Berikan izin untuk permintaan dari instans Amazon EC2

Contoh berikut kebijakan Amazon SQS memberikan akses ke permintaan yang berasal dari instans Amazon EC2. Contoh ini dibangun di atas contoh Contoh 2: Berikan izin ke satu atau beberapa akun "": membatasi akses sebelum 30 Juni 2009 pukul 12 siang (UTC), ini membatasi akses ke rentang IP. `203.0.113.0/24` Penting untuk menulis kebijakan ini dan mengunggahnya ke Amazon SQS menggunakan `SetQueueAttributes`tindakan karena `AddPermission`tindakan tidak mengizinkan penetapan pembatasan alamat IP saat memberikan akses ke antrian.

```
{
   "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"
         }
      }
   }]
}
```

Contoh 4: Tolak akses ke akun tertentu

Contoh berikut kebijakan Amazon SQS menolak Akun AWS akses tertentu ke antrian Anda. Contoh ini dibangun di atas contoh Contoh 1: Berikan izin ke satu akun "": menolak akses ke yang ditentukan. Akun AWS Penting untuk menulis kebijakan ini dan mengunggahnya ke Amazon SQS menggunakan `SetQueueAttributes`tindakan karena `AddPermission`tindakan tidak mengizinkan penolakan akses ke antrian (hanya memungkinkan pemberian akses ke antrian).

```
{
   "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"
   }]
}
```

Contoh 5: Tolak akses jika bukan dari titik akhir VPC

Contoh kebijakan Amazon SQS berikut membatasi akses kequeue1: 111122223333 `ReceiveMessage`hanya dapat melakukan `SendMessage`tindakan dan dari ID titik akhir VPC (ditentukan menggunakan kondisi). `vpce-1a2b3c4d` `aws:sourceVpce` Untuk informasi selengkapnya, lihat Titik akhir Amazon Virtual Private Cloud untuk Amazon SQS.

> **Note**
>
> - `aws:sourceVpce`Kondisi ini tidak memerlukan ARN untuk sumber daya titik akhir VPC, hanya ID titik akhir VPC.
> - Anda dapat memodifikasi contoh berikut untuk membatasi semua tindakan ke titik akhir VPC tertentu dengan menolak semua `sqs:*` tindakan Amazon SQS () dalam pernyataan

kedua. Namun, pernyataan kebijakan tersebut akan menetapkan bahwa semua tindakan (termasuk tindakan administratif yang diperlukan untuk mengubah izin antrian) harus dilakukan melalui titik akhir VPC tertentu yang ditentukan dalam kebijakan, yang berpotensi mencegah pengguna memodifikasi izin antrian di masa mendatang.

```
{
   "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"
            }
         }
      }
   ]
}
```

Menggunakan kredensi keamanan sementara dengan Amazon SQS

Selain membuat pengguna dengan kredensi keamanan mereka sendiri, IAM juga memungkinkan Anda untuk memberikan kredensi keamanan sementara kepada pengguna mana pun, memungkinkan pengguna untuk mengakses layanan dan sumber daya Anda. AWS Anda dapat mengelola pengguna yang memiliki Akun AWS. Anda juga dapat mengelola pengguna untuk sistem Anda yang tidak memiliki Akun AWS (pengguna federasi). Selain itu, aplikasi yang Anda buat untuk mengakses AWS sumber daya Anda juga dapat dianggap sebagai “pengguna.”

Anda dapat menggunakan kredensyal keamanan sementara ini untuk membuat permintaan ke Amazon SQS. Pustaka API menghitung nilai tanda tangan yang diperlukan menggunakan kredensial tersebut untuk mengautentikasi permintaan Anda. Jika Anda mengirim permintaan menggunakan kredensi kedaluwarsa, Amazon SQS menolak permintaan tersebut.

> **Note**
>
> Anda tidak dapat menetapkan kebijakan berdasarkan kredensional sementara.

Prasyarat

1. Gunakan IAM untuk membuat kredenal keamanan sementara:
   - Token keamanan
   - Access key ID
   - Kunci Akses Rahasia
2. Siapkan string Anda untuk masuk dengan ID Kunci Akses sementara dan token keamanan.
3. Gunakan Kunci Akses Rahasia sementara alih-alih Kunci Akses Rahasia Anda sendiri untuk menandatangani permintaan API Kueri Anda.

> **Note**
>
> Saat Anda mengirimkan permintaan Query API yang ditandatangani, gunakan ID Kunci Akses sementara, bukan ID Kunci Akses Anda sendiri dan untuk menyertakan token keamanan. Untuk informasi selengkapnya tentang dukungan IAM untuk kredensil keamanan sementara, lihat Memberikan Akses Sementara ke AWS Sumber Daya Anda di Panduan Pengguna IAM.

Untuk memanggil tindakan API Kueri Amazon SQS menggunakan kredenal keamanan sementara

1. Minta token keamanan sementara menggunakan AWS Identity and Access Management. Untuk informasi selengkapnya, lihat Membuat Kredensial Keamanan Sementara untuk Mengaktifkan Akses bagi Pengguna IAM di Panduan Pengguna IAM.

   IAM mengembalikan token keamanan, ID Kunci Akses, dan Kunci Akses Rahasia.
2. Siapkan kueri Anda menggunakan ID Kunci Akses sementara alih-alih ID Kunci Akses Anda sendiri dan sertakan token keamanan. Tanda tangani permintaan Anda menggunakan Kunci Akses Rahasia sementara, bukan milik Anda sendiri.
3. Kirim string kueri yang ditandatangani dengan ID Kunci Akses sementara dan token keamanan.

   Contoh berikut menunjukkan cara menggunakan kredenal keamanan sementara untuk mengautentikasi permintaan Amazon SQS. Struktur *AUTHPARAMS*tergantung pada tanda tangan permintaan API. Untuk informasi selengkapnya, lihat Menandatangani Permintaan AWS API di Referensi Umum 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
```

   Contoh berikut menggunakan kredenal keamanan sementara untuk mengirim dua pesan menggunakan tindakan. `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
```

Manajemen akses untuk antrian Amazon SQS terenkripsi dengan kebijakan hak istimewa paling sedikit

Anda dapat menggunakan Amazon SQS untuk bertukar data sensitif antar aplikasi dengan menggunakan enkripsi sisi server (SSE) yang terintegrasi dengan (KMS).AWS Key Management Service Dengan integrasi Amazon SQS dan AWS KMS, Anda dapat mengelola kunci yang melindungi Amazon SQS secara terpusat, serta kunci yang melindungi sumber daya Anda yang lain. AWS

Beberapa AWS layanan dapat bertindak sebagai sumber peristiwa yang mengirim acara ke Amazon SQS. Untuk mengaktifkan sumber peristiwa untuk mengakses antrean Amazon SQS terenkripsi, Anda perlu mengonfigurasi antrian dengan kunci yang dikelola pelanggan. AWS KMS Kemudian, gunakan kebijakan kunci untuk mengizinkan layanan menggunakan metode AWS KMS API yang diperlukan. Layanan ini juga memerlukan izin untuk mengautentikasi akses untuk mengaktifkan antrian untuk mengirim acara. Anda dapat mencapai ini dengan menggunakan kebijakan Amazon SQS, yang merupakan kebijakan berbasis sumber daya yang dapat Anda gunakan untuk mengontrol akses ke antrean Amazon SQS dan datanya.

Bagian berikut memberikan informasi tentang cara mengontrol akses ke antrean Amazon SQS terenkripsi melalui kebijakan Amazon SQS dan kebijakan utama. AWS KMS Kebijakan dalam panduan ini akan membantu Anda mencapai hak istimewa paling sedikit.

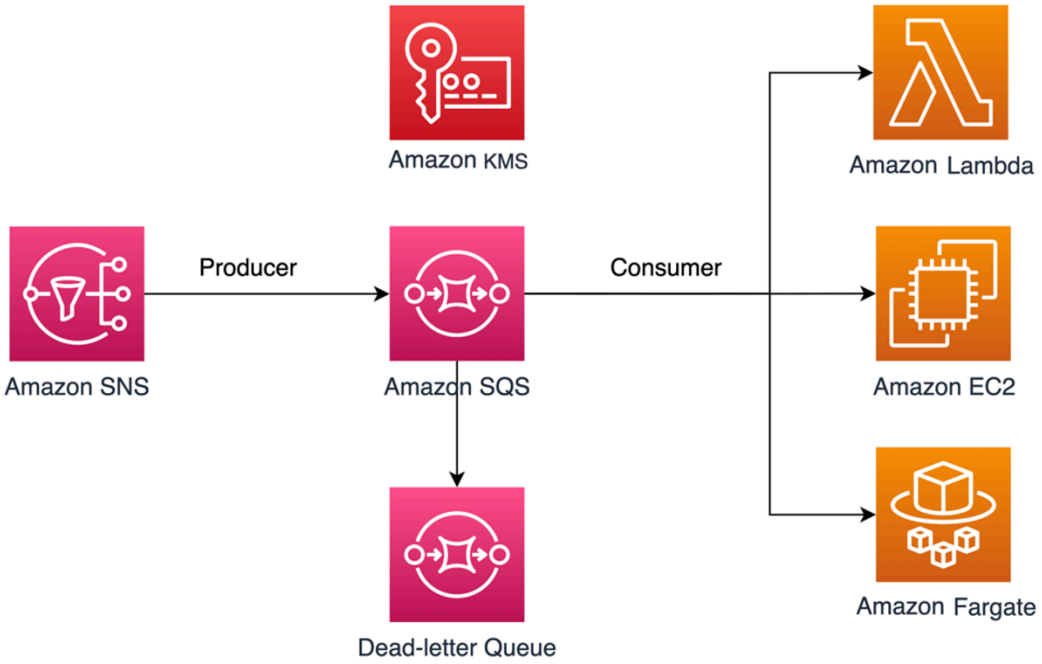
Panduan ini juga menjelaskan bagaimana kebijakan berbasis sumber daya mengatasi masalah wakil yang membingungkan dengan menggunakan kunci konteks kondisi IAM `aws:SourceArn`,, `aws:SourceAccount`dan global. `aws:PrincipalOrgID`

Topik

- Gambaran Umum
- Kebijakan kunci hak istimewa paling sedikit untuk Amazon SQS
- Pernyataan kebijakan Amazon SQS untuk antrian huruf mati
- Mencegah masalah wakil lintas layanan yang membingungkan
- Gunakan IAM Access Analyzer untuk meninjau akses lintas akun

Gambaran Umum

Dalam topik ini, kami akan memandu Anda melalui kasus penggunaan umum untuk menggambarkan bagaimana Anda dapat membangun kebijakan kunci dan kebijakan antrian Amazon SQS. Kasus penggunaan ini ditunjukkan pada gambar berikut.



Dalam contoh ini, produsen pesan adalah topik Amazon Simple Notification Service (SNS), yang dikonfigurasi untuk meng-fanout pesan ke antrean Amazon SQS terenkripsi Anda. Konsumen pesan adalah layanan komputasi, seperti AWS Lambdafungsi, instans Amazon Elastic Compute Cloud (EC2), atau container. AWS Fargate Antrian Amazon SQS Anda kemudian dikonfigurasi untuk mengirim pesan gagal ke Dead-Letter Queue (DLQ). Ini berguna untuk men-debug aplikasi atau sistem pesan Anda karena DLQ memungkinkan Anda mengisolasi pesan yang tidak dikonsumsi untuk menentukan mengapa pemrosesannya tidak berhasil. Dalam solusi yang ditentukan dalam topik ini, layanan komputasi seperti fungsi Lambda digunakan untuk memproses pesan yang disimpan dalam antrean Amazon SQS. Jika konsumen pesan berada di cloud pribadi virtual (VPC), pernyataan `DenyReceivingIfNotThroughVPCE`kebijakan yang disertakan dalam panduan ini memungkinkan Anda membatasi penerimaan pesan ke VPC tertentu.

> **Note**
>
> Panduan ini hanya berisi izin IAM yang diperlukan dalam bentuk pernyataan kebijakan. Untuk membuat kebijakan, Anda perlu menambahkan pernyataan ke kebijakan Amazon SQS atau kebijakan utama AWS KMS Anda. Panduan ini tidak memberikan petunjuk tentang cara

membuat antrean Amazon SQS atau kuncinya. AWS KMS Untuk petunjuk tentang cara membuat sumber daya ini, lihat Membuat antrian Amazon SQS dan Membuat kunci. Kebijakan Amazon SQS yang ditentukan dalam panduan ini tidak mendukung penggerak ulang pesan secara langsung ke antrean Amazon SQS yang sama atau berbeda.

Kebijakan kunci hak istimewa paling sedikit untuk Amazon SQS

Di bagian ini, kami menjelaskan izin hak istimewa terkecil yang diperlukan untuk kunci yang dikelola pelanggan yang Anda gunakan AWS KMS untuk mengenkripsi antrean Amazon SQS Anda. Dengan izin ini, Anda dapat membatasi akses hanya ke entitas yang dituju sambil menerapkan hak istimewa paling sedikit. Kebijakan utama harus terdiri dari pernyataan kebijakan berikut, yang kami jelaskan secara rinci di bawah ini:

- Berikan izin administrator ke kunci AWS KMS
- Berikan akses hanya-baca ke metadata kunci
- Berikan izin Amazon SNS KMS ke Amazon SNS untuk mempublikasikan pesan ke antrian
- Memungkinkan konsumen untuk mendekripsi pesan dari antrian

Berikan izin administrator ke kunci AWS KMS

Untuk membuat AWS KMS kunci, Anda perlu memberikan izin AWS KMS administrator ke peran IAM yang Anda gunakan untuk menyebarkan kunci. AWS KMS Izin administrator ini didefinisikan dalam pernyataan `AllowKeyAdminPermissions` kebijakan berikut. Saat menambahkan pernyataan ini ke kebijakan AWS KMS kunci, pastikan untuk mengganti <admin-role ARN>dengan Amazon Resource Name (ARN) peran IAM yang digunakan untuk menyebarkan AWS KMS kunci, mengelola kunci, atau keduanya AWS KMS . Ini bisa berupa peran IAM dari pipeline penerapan Anda, atau peran administrator untuk organisasi Anda di 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**
>
> Dalam kebijakan AWS KMS kunci, nilai `Resource` elemen harus`*`, yang berarti " AWS KMS kunci ini". Tanda bintang (`*`) mengidentifikasi AWS KMS kunci yang dilampirkan kebijakan kunci.

Berikan akses hanya-baca ke metadata kunci

Untuk memberikan akses hanya-baca peran IAM lainnya ke metadata kunci Anda, tambahkan `AllowReadAccessToKeyMetaData` pernyataan tersebut ke kebijakan kunci Anda. Misalnya, pernyataan berikut memungkinkan Anda mencantumkan semua AWS KMS kunci di akun Anda untuk tujuan audit. Pernyataan ini memberikan akses read-only kepada pengguna AWS root ke metadata kunci. Oleh karena itu, setiap prinsipal IAM di akun dapat memiliki akses ke metadata kunci ketika kebijakan berbasis identitas mereka memiliki izin yang tercantum dalam pernyataan berikut:, dan. `kms:Describe*` `kms:Get*` `kms:List*` Pastikan untuk mengganti <account-ID>dengan informasi Anda sendiri.

```
{
  "Sid": "AllowReadAcesssToKeyMetaData",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "arn:aws:iam::<accountID>:root"
```

```
    ]
  },
  "Action": [
    "kms:Describe*",
    "kms:Get*",
    "kms:List*"
  ],
  "Resource": "*"
}
```

Berikan izin Amazon SNS KMS ke Amazon SNS untuk mempublikasikan pesan ke antrian

Untuk mengizinkan topik Amazon SNS Anda memublikasikan pesan ke antrean Amazon SQS terenkripsi, tambahkan `AllowSNSToSendToSQS` pernyataan kebijakan ke kebijakan utama Anda. Pernyataan ini memberikan izin Amazon SNS untuk menggunakan AWS KMS kunci untuk mempublikasikan ke antrian Amazon SQS Anda. Pastikan untuk mengganti <account-ID>dengan informasi Anda sendiri.

> **Note**
>
> `Condition`Dalam pernyataan membatasi akses hanya ke layanan Amazon SNS di akun yang sama AWS .

```
{
  "Sid": "AllowSNSToSendToSQS",
  "Effect": "Allow",
  "Principal": {
    "Service": [
      "sns.amazonaws.com"
    ]
  },
  "Action": [
    "kms:GenerateDataKey",
    "kms:Decrypt"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "aws:SourceAccount": "<account-id>"
    }
```

```
  }
}
```

Memungkinkan konsumen untuk mendekripsi pesan dari antrian

`AllowConsumersToReceiveFromTheQueue`Pernyataan berikut memberi konsumen pesan Amazon SQS izin yang diperlukan untuk mendekripsi pesan yang diterima dari antrian Amazon SQS terenkripsi. Saat Anda melampirkan pernyataan kebijakan, ganti peran *`runtime <consumer ARN>dengan ARN peran`* runtime IAM dari konsumen pesan.

```
{
  "Sid": "AllowConsumersToReceiveFromTheQueue",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "<consumer's execution role ARN>"
    ]
  },
  "Action": [
    "kms:Decrypt"
  ],
  "Resource": "*"
}
```

Kebijakan Amazon SQS dengan hak istimewa paling sedikit

Bagian ini memandu Anda melalui kebijakan antrian Amazon SQS dengan hak istimewa paling sedikit untuk kasus penggunaan yang dicakup oleh panduan ini (misalnya, Amazon SNS ke Amazon SQS). Kebijakan yang ditetapkan dirancang untuk mencegah akses yang tidak diinginkan dengan menggunakan campuran keduanya `Deny` dan `Allow` pernyataan. `Allow`Pernyataan memberikan akses ke entitas atau entitas yang dimaksud. `Deny`Pernyataan tersebut mencegah entitas lain yang tidak diinginkan mengakses antrean Amazon SQS, sementara mengecualikan entitas yang dimaksud dalam kondisi kebijakan.

Kebijakan Amazon SQS mencakup pernyataan berikut, yang kami jelaskan secara rinci di bawah ini:

- Batasi izin manajemen Amazon SQS
- Batasi tindakan antrian Amazon SQS dari organisasi yang ditentukan
- Berikan izin Amazon SQS kepada konsumen
- Menegakkan enkripsi dalam perjalanan

- Batasi transmisi pesan ke topik Amazon SNS tertentu
- (Opsional) Batasi penerimaan pesan ke titik akhir VPC tertentu

Batasi izin manajemen Amazon SQS

Pernyataan `RestrictAdminQueueActions` kebijakan berikut membatasi izin pengelolaan Amazon SQS hanya untuk peran atau peran IAM yang Anda gunakan untuk menerapkan antrean, mengelola antrian, atau keduanya. Pastikan untuk mengganti <placeholder values>dengan informasi Anda sendiri. Tentukan ARN peran IAM yang digunakan untuk menerapkan antrean Amazon SQS, serta ARN dari setiap peran administrator yang harus memiliki izin pengelolaan 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>"
      ]
    }
  }
}
```

Batasi tindakan antrian Amazon SQS dari organisasi yang ditentukan

Untuk membantu melindungi sumber daya Amazon SQS Anda dari akses eksternal (akses oleh entitas di luar AWS organisasi Anda), gunakan pernyataan berikut. Pernyataan ini membatasi akses antrian Amazon SQS ke organisasi yang Anda tentukan di. `Condition` <org-id>Pastikan untuk mengganti <SQS queue ARN>dengan ARN peran IAM yang digunakan untuk menerapkan antrean Amazon SQS; dan, dengan ID organisasi Anda.

```
{
  "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>"
      ]
    }
  }
}
```

Berikan izin Amazon SQS kepada konsumen

Untuk menerima pesan dari antrian Amazon SQS, Anda perlu memberikan izin yang diperlukan kepada konsumen pesan. Pernyataan kebijakan berikut memberi konsumen, yang Anda tentukan, izin yang diperlukan untuk menggunakan pesan dari antrian Amazon SQS. Saat menambahkan pernyataan ke kebijakan Amazon SQS Anda, pastikan untuk mengganti peran runtime *IAM <consumer ARN>dengan ARN dari peran runtime* IAM yang digunakan oleh konsumen; dan, <SQS queue ARN>dengan ARN dari peran IAM yang digunakan untuk menerapkan antrean 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>"
}
```

Untuk mencegah entitas lain menerima pesan dari antrean Amazon SQS, tambahkan `DenyOtherConsumersFromReceiving` pernyataan ke kebijakan antrian Amazon SQS. Pernyataan ini membatasi konsumsi pesan kepada konsumen yang Anda tetapkan—tidak memungkinkan konsumen lain untuk memiliki akses, bahkan ketika izin identitas mereka akan memberi mereka akses. Pastikan untuk mengganti <SQS queue ARN>dan *<peran runtime konsumen ARN>*dengan informasi Anda sendiri.

```
{
  "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>"
    }
  }
}
```

Menegakkan enkripsi dalam perjalanan

Pernyataan `DenyUnsecureTransport` kebijakan berikut memberlakukan konsumen dan produsen untuk menggunakan saluran aman (koneksi TLS) untuk mengirim dan menerima pesan dari antrian Amazon SQS. Pastikan untuk mengganti <SQS queue ARN>dengan ARN dari peran IAM yang digunakan untuk menyebarkan antrian Amazon SQS.

```
{
  "Sid": "DenyUnsecureTransport",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": [
    "sqs:ReceiveMessage",
    "sqs:SendMessage"
  ],
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "Bool": {
      "aws:SecureTransport": "false"
    }
  }
}
```

Batasi transmisi pesan ke topik Amazon SNS tertentu

Pernyataan `AllowSNSToSendToTheQueue` kebijakan berikut memungkinkan topik Amazon SNS yang ditentukan untuk mengirim pesan ke antrean Amazon SQS. <SNS topic ARN>Pastikan untuk mengganti <SQS queue ARN>dengan ARN peran IAM yang digunakan untuk menyebarkan antrian Amazon SQS; dan, dengan topik Amazon SNS ARN.

```
{
  "Sid": "AllowSNSToSendToTheQueue",
  "Effect": "Allow",
  "Principal": {
    "Service": "sns.amazonaws.com"
  },
  "Action": "sqs:SendMessage",
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "ArnLike": {
      "aws:SourceArn": "<SNS topic ARN>"
    }
  }
}
```

Pernyataan `DenyAllProducersExceptSNSFromSending` kebijakan berikut mencegah produsen lain mengirim pesan ke antrian. Ganti <SQS queue ARN>dan <SNS topic ARN>dengan informasi Anda sendiri.

```
{
  "Sid": "DenyAllProducersExceptSNSFromSending",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": "sqs:SendMessage",
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "ArnNotLike": {
      "aws:SourceArn": "<SNS topic ARN>"
    }
  }
}
```

(Opsional) Batasi penerimaan pesan ke titik akhir VPC tertentu

Untuk membatasi penerimaan pesan hanya ke titik akhir VPC tertentu, tambahkan pernyataan kebijakan berikut ke kebijakan antrian Amazon SQS Anda. Pernyataan ini mencegah konsumen pesan menerima pesan dari antrian kecuali pesan tersebut berasal dari titik akhir VPC yang diinginkan. <vpce_id>Ganti <SQS queue ARN>dengan ARN dari peran IAM yang digunakan untuk menyebarkan antrian Amazon SQS; dan dengan ID titik akhir VPC.

```
{
  "Sid": "DenyReceivingIfNotThroughVPCE",
  "Effect": "Deny",
  "Principal": "*",
  "Action": [
    "sqs:ReceiveMessage"
  ],
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "StringNotEquals": {
      "aws:sourceVpce": "<vpce id>"
    }
  }
```

```
}
```

Pernyataan kebijakan Amazon SQS untuk antrian huruf mati

Tambahkan pernyataan kebijakan berikut, yang diidentifikasi oleh ID pernyataan mereka, ke kebijakan akses DLQ Anda:

- `RestrictAdminQueueActions`
- `DenyQueueActionsOutsideOrg`
- `AllowConsumersToReceiveFromTheQueue`
- `DenyOtherConsumersFromReceiving`
- `DenyUnsecureTransport`

Selain menambahkan pernyataan kebijakan sebelumnya ke kebijakan akses DLQ Anda, Anda juga harus menambahkan pernyataan untuk membatasi transmisi pesan ke antrian Amazon SQS, seperti yang dijelaskan di bagian berikut.

Batasi transmisi pesan ke antrian Amazon SQS

Untuk membatasi akses hanya antrian Amazon SQS dari akun yang sama, tambahkan pernyataan kebijakan `DenyAnyProducersExceptSQS` berikut ke kebijakan antrian DLQ. Pernyataan ini tidak membatasi transmisi pesan ke antrian tertentu karena Anda perlu menerapkan DLQ sebelum membuat antrian utama, sehingga Anda tidak akan tahu Amazon SQS ARN saat membuat DLQ. Jika Anda perlu membatasi akses ke hanya satu antrian Amazon SQS, ubah `aws:SourceArn` di dengan ARN dari antrian sumber Amazon SQS Anda saat Anda mengetahuinya. `Condition`

```
{
  "Sid": "DenyAnyProducersExceptSQS",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": "sqs:SendMessage",
  "Resource": "<SQS DLQ ARN>",
  "Condition": {
    "ArnNotLike": {
      "aws:SourceArn": "arn:aws:sqs:<region>:<account-id>:*"
    }
  }
```

```
}
```

> **Important**
>
> Kebijakan antrean Amazon SQS yang ditentukan dalam panduan ini tidak membatasi `sqs:PurgeQueue` tindakan ke peran atau peran IAM tertentu. `sqs:PurgeQueue`Tindakan ini memungkinkan Anda untuk menghapus semua pesan dalam antrean Amazon SQS. Anda juga dapat menggunakan tindakan ini untuk membuat perubahan pada format pesan tanpa mengganti antrean Amazon SQS. Saat men-debug aplikasi, Anda dapat menghapus antrean Amazon SQS untuk menghapus pesan yang berpotensi salah. Saat menguji aplikasi, Anda dapat mengarahkan volume pesan tinggi melalui antrian Amazon SQS dan kemudian membersihkan antrian untuk memulai yang baru sebelum memasuki produksi. Alasan untuk tidak membatasi tindakan ini ke peran tertentu adalah bahwa peran ini mungkin tidak diketahui saat menerapkan antrian Amazon SQS. Anda perlu menambahkan izin ini ke kebijakan berbasis identitas peran agar dapat membersihkan antrian.

Mencegah masalah wakil lintas layanan yang membingungkan

Masalah deputi yang membingungkan adalah masalah keamanan di mana entitas yang tidak memiliki izin untuk melakukan tindakan dapat memaksa entitas yang lebih istimewa untuk melakukan tindakan. Untuk mencegah hal ini, AWS sediakan alat yang membantu Anda melindungi akun Anda jika Anda memberi pihak ketiga (dikenal sebagai lintas akun) atau AWS layanan lain (dikenal sebagai lintas layanan) akses ke sumber daya di akun Anda. Pernyataan kebijakan di bagian ini dapat membantu Anda mencegah masalah wakil lintas layanan yang membingungkan.

Peniruan identitas lintas layanan dapat terjadi ketika satu layanan (layanan yang dipanggil) memanggil layanan lain (layanan yang dipanggil). Layanan panggilan dapat dimanipulasi untuk menggunakan izinnya untuk bertindak atas sumber daya pelanggan lain dengan cara yang seharusnya tidak memiliki izin untuk mengakses. Untuk membantu melindungi dari masalah ini, kebijakan berbasis sumber daya yang ditentukan dalam posting ini menggunakan kunci konteks kondisi IAM `aws:SourceArn`, `aws:SourceAccount`, dan `aws:PrincipalOrgID`global. Ini membatasi izin yang dimiliki layanan untuk sumber daya tertentu, akun tertentu, atau organisasi tertentu di AWS Organizations.

Gunakan IAM Access Analyzer untuk meninjau akses lintas akun

Anda dapat menggunakan AWS IAM Access Analyzer untuk meninjau kebijakan antrean Amazon SQS AWS KMS dan kebijakan kunci serta memberi tahu Anda saat antrian Amazon SQS atau kunci

AWS KMS memberikan akses ke entitas eksternal. IAM Access Analyzer membantu mengidentifikasi sumber daya di organisasi dan akun Anda yang dibagikan dengan entitas di luar zona kepercayaan. Zona kepercayaan ini dapat berupa AWS akun atau AWS organisasi dalam Organizations yang Anda tentukan saat mengaktifkan IAM Access Analyzer.

IAM Access Analyzer mengidentifikasi sumber daya yang dibagikan dengan prinsipal eksternal dengan menggunakan penalaran berbasis logika untuk menganalisis kebijakan berbasis sumber daya di lingkungan Anda. AWS Untuk setiap contoh sumber daya yang dibagikan di luar zona kepercayaan Anda, Access Analyzer menghasilkan temuan. Temuan mencakup informasi tentang akses dan prinsip eksternal yang diberikan kepadanya. Tinjau temuan untuk menentukan apakah akses dimaksudkan dan aman, atau apakah akses tidak diinginkan dan risiko keamanan. Untuk akses yang tidak diinginkan, tinjau kebijakan yang terpengaruh dan perbaiki. Lihat posting blog ini untuk informasi lebih lanjut tentang bagaimana AWS IAM Access Analyzer mengidentifikasi akses yang tidak diinginkan ke sumber daya Anda. AWS

Untuk informasi selengkapnya tentang AWS IAM Access Analyzer, lihat dokumentasi AWS IAM Access Analyzer.

Izin Amazon SQS API: Tindakan dan referensi sumber daya

Saat menyiapkan Kontrol akses dan menulis kebijakan izin yang dapat dilampirkan ke identitas IAM, Anda dapat menggunakan tabel berikut sebagai referensi. Daftar mencakup setiap tindakan Layanan Antrian Sederhana Amazon, tindakan terkait yang dapat Anda berikan izin untuk melakukan tindakan, dan AWS sumber daya yang dapat Anda berikan izin.

Tentukan tindakan di `Action` bidang kebijakan, dan nilai sumber daya di `Resource` bidang kebijakan. Untuk menentukan tindakan, gunakan `sqs:` awalan diikuti dengan nama tindakan (misalnya,`sqs:CreateQueue`).

Saat ini, Amazon SQS mendukung kunci konteks kondisi global yang tersedia di IAM.

Amazon Simple Queue Service API dan izin yang diperlukan untuk tindakan

AddPermission

Tindakan: `sqs:AddPermission`

Sumber daya: `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

ChangeMessageVisibilitas

Tindakan: `sqs:ChangeMessageVisibility`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

ChangeMessageVisibilityBatch

Tindakan: `sqs:ChangeMessageVisibilityBatch`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

CreateQueue

Tindakan: `sqs:CreateQueue`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

DeleteMessage

Tindakan: `sqs:DeleteMessage`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

DeleteMessageBatch

Tindakan: `sqs:DeleteMessageBatch`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

DeleteQueue

Tindakan: `sqs:DeleteQueue`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

GetQueueAtribut

Tindakan: `sqs:GetQueueAttributes`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

GetQueueUrl

Tindakan: `sqs:GetQueueUrl`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

ListDeadLetterSourceAntrian

Tindakan: `sqs:ListDeadLetterSourceQueues`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

ListQueues

Tindakan: `sqs:ListQueues`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

ListQueueTag

Tindakan: `sqs:ListQueueTags`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

PurgeQueue

Tindakan: `sqs:PurgeQueue`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

ReceiveMessage

Tindakan: `sqs:ReceiveMessage`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

RemovePermission

Tindakan: `sqs:RemovePermission`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

SendMessagedan SendMessageBatch

Tindakan: `sqs:SendMessage`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

SetQueueAtribut

Tindakan: `sqs:SetQueueAttributes`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

TagQueue

Tindakan: `sqs:TagQueue`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

UntagQueue

Tindakan: `sqs:UntagQueue`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

Pencatatan dan pemantauan di Amazon SQS

Bagian ini memberikan informasi tentang opsi pencatatan dan pemantauan untuk Amazon SQS, termasuk cara menggunakan CloudTrail untuk menangkap panggilan API, dan CloudWatch metrik untuk mendapatkan wawasan tentang aktivitas dan kinerja antrian.

Topik

- Pencatatan panggilan API Amazon SQS menggunakan AWS CloudTrail
- Memantau antrian Amazon SQS menggunakan CloudWatch

Pencatatan panggilan API Amazon SQS menggunakan AWS CloudTrail

Amazon SQS terintegrasi dengan AWS CloudTrail untuk merekam panggilan Amazon SQS dari pengguna, peran, atau layanan. AWS CloudTrail menangkap panggilan API yang terkait dengan standar Amazon SQS dan antrian FIFO sebagai peristiwa, termasuk interaksi yang dimulai melalui konsol Amazon SQS serta secara terprogram melalui panggilan ke Amazon SQS API.

Topik

- Informasi Amazon SQS di CloudTrail
- Acara manajemen di CloudTrail
- Peristiwa data di CloudTrail
- Contoh: acara CloudTrail manajemen untuk Amazon SQS
- Contoh: peristiwa CloudTrail data untuk Amazon SQS

Informasi Amazon SQS di CloudTrail

CloudTrail diaktifkan secara default saat Anda membuat AWS akun. Ketika aktivitas peristiwa Amazon SQS yang didukung terjadi, aktivitas tersebut direkam dalam suatu CloudTrail peristiwa, bersama dengan peristiwa AWS layanan lainnya, dalam riwayat acara. Anda dapat melihat, mencari,

dan mengunduh acara terbaru untuk AWS akun Anda. Untuk informasi selengkapnya, lihat Melihat CloudTrail Acara dengan Riwayat Acara di Panduan AWS CloudTrail Pengguna.

Amazon SQS API yang memanggil operasi manajemen antrian, seperti `AddPermission` dikategorikan sebagai peristiwa manajemen dan masuk secara default. CloudTrail Amazon SQS API yang merupakan operasi volume tinggi yang dilakukan pada antrean Amazon SQS, seperti dikategorikan `SendMessage` sebagai peristiwa data dan dicatat setelah Anda ikut serta. CloudTrail

Dengan menggunakan informasi yang CloudTrail dikumpulkan, Anda dapat mengidentifikasi permintaan khusus ke Amazon SQS API, alamat IP atau identitas pemohon, serta tanggal dan waktu permintaan. Jika mengonfigurasi CloudTrail jejak, Anda dapat terus mengirimkan CloudTrail peristiwa ke bucket Amazon S3 dengan pengiriman opsional ke Amazon CloudWatch Log dan. AWS EventBridge Jika Anda tidak mengonfigurasi jejak, Anda hanya dapat melihat riwayat acara acara acara manajemen dalam acara di CloudTrail konsol. Untuk informasi selengkapnya, lihat Gambaran Umum Pembuatan Jejak di Panduan Pengguna AWS CloudTrail.

Acara manajemen di CloudTrail

Amazon SQS mencatat tindakan API berikut sebagai peristiwa manajemen:

- `AddPermission`
- `CreateQueue`
- `CancelMessageMoveTask`
- `DeleteQueue`
- `ListMessageMoveTasks`
- `PurgeQueue`
- `RemovePermission`
- `SetQueueAttributes`
- `StartMessageMoveTask`
- `TagQueue`
- `UntagQueue`

API Amazon SQS berikut tidak didukung untuk CloudTrail pencatatan:

- `GetQueueAttributes`
- `GetQueueUrl`

- `ListDeadLetterSourceQueues`
- `ListQueueTags`
- `ListQueues`

Peristiwa data di CloudTrail

Peristiwa data memberikan informasi tentang operasi sumber daya yang dilakukan pada atau di sumber daya, seperti mengirim atau menerima pesan Amazon SQS ke dan dari antrian Amazon SQS. Peristiwa data adalah aktivitas volume tinggi yang CloudTrail tidak masuk secara default. Anda dapat mengaktifkan pencatatan tindakan API peristiwa data untuk antrean SQS Anda dengan menggunakan CloudTrail API. Untuk informasi selengkapnya, lihat Mencatat peristiwa data dalam AWS CloudTrail Panduan Pengguna.

Dengan CloudTrail, Anda dapat menggunakan penyeleksi peristiwa lanjutan untuk memutuskan aktivitas API Amazon SQS mana yang dicatat dan direkam. Untuk mencatat peristiwa data Amazon SQS, Anda harus menyertakan jenis sumber daya. `AWS::SQS::Queue` Setelah ini diatur, Anda dapat memperbaiki preferensi logging Anda lebih lanjut dengan memilih peristiwa data tertentu untuk direkam, seperti menggunakan `eventName` filter untuk melacak `SendMessage` peristiwa. Untuk informasi selengkapnya, lihat `AdvancedEventSelector` di dalam Referensi API AWS CloudTrail .

Peristiwa data Amazon SQS:

- `SendMessage`
- `SendMessageBatch`
- `ReceiveMessage`
- `DeleteMessage`
- `DeleteMessageBatch`
- `ChangeMessageVisibility`
- `ChangeMessageVisibilityBatch`

Biaya tambahan berlaku untuk peristiwa data. Untuk informasi selengkapnya, silakan lihat Harga AWS CloudTrail.

Contoh: acara CloudTrail manajemen untuk Amazon SQS

Contoh berikut menunjukkan entri CloudTrail log untuk API yang didukung:

AddPermission

Contoh berikut menunjukkan entri CloudTrail log untuk panggilan `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

Contoh berikut menunjukkan entri CloudTrail log untuk panggilan `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

Contoh berikut menunjukkan entri CloudTrail log untuk panggilan `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

Contoh berikut menunjukkan entri CloudTrail log untuk panggilan `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

Contoh berikut menunjukkan entri CloudTrail log untuk`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"
     }
```

```
    ]
  }
```

Contoh: peristiwa CloudTrail data untuk Amazon SQS

Berikut ini adalah contoh CloudTrail peristiwa khusus untuk API peristiwa data Amazon SQS:

SendMessage

Contoh berikut menunjukkan peristiwa CloudTrail data untuk`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"
  }
```


ReceiveMessage

Contoh berikut menunjukkan peristiwa CloudTrail data untuk`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"
  }
}
```

DeleteMessageBatch

Contoh berikut menunjukkan peristiwa CloudTrail data untuk`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/
hpb4RbXHPZ9tLBVleECbChQIE5NtaDuoZhZPOkTy0eN46EyRR4jXDzE3AlkbPlX1mA9f2fUuTrXx8aeCoCA3I3woNg3f
hlLS94tjAZqV2krc4BaC2pYgjyHWcW019HwIV8T/bjNMIeZoQwOM5V
+o9vHPfewz5QGr5SKpDo7uE7Umyk5n5CJZvcn1efp/
mrwtaCIb9M7cCQUYcZm2ZmZDnIO9XpGTAi3m2dQ0M83pnNh0nvDfpkHpoa+hX1TrUmxCupCWHJwA8HFJ1O/
CCJsodMNFthLBA9S57dkBZCsw41G8jAmgQ0MkvZ0UL5mg0OFQQd1Yrw0zvthjCgiwdzn0yXoMzxIZMBxkY14E4nVVZ7N
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"
  }
}
```

ChangeMessageVisibilityBatch

Contoh berikut menunjukkan peristiwa CloudTrail data untuk`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/
```

```
abV3szqVWXOLZR4PwX8zZkWVQGNCNnY2q2lGCG586F8QwvrOFYoXNwB8ymd1t77e1PDPknq1Io3JFuzkEsndkkETy4fV
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"
  }
}
```

Memantau antrian Amazon SQS menggunakan CloudWatch

Amazon SQS dan Amazon CloudWatch terintegrasi sehingga Anda dapat menggunakan CloudWatch untuk melihat dan menganalisis metrik untuk antrian Amazon SQS Anda. Anda dapat melihat dan menganalisis metrik antrian dari konsol Amazon SQS, konsol, menggunakan, atau menggunakan API AWS CLI. CloudWatch CloudWatch Anda juga dapat mengatur CloudWatch alarm untuk metrik Amazon SQS.

CloudWatch metrik untuk antrian Amazon SQS Anda secara otomatis dikumpulkan dan didorong ke CloudWatch interval satu menit. Metrik ini dikumpulkan pada semua antrian yang memenuhi CloudWatch pedoman untuk aktif. CloudWatchmenganggap antrian aktif hingga enam jam jika berisi pesan apa pun, atau jika ada tindakan yang mengaksesnya.

Ketika antrean Amazon SQS tidak aktif selama lebih dari enam jam, layanan Amazon SQS dianggap tertidur dan berhenti mengirimkan metrik ke layanan. CloudWatch Data yang hilang, atau data yang mewakili nol, tidak dapat divisualisasikan dalam CloudWatch metrik untuk Amazon SQS selama periode waktu antrian Amazon SQS Anda tidak aktif.

> **Note**
>
> - Antrian Amazon SQS dapat diaktifkan saat pengguna memanggil API terhadap antrian tidak diotorisasi, dan permintaan gagal.
> - Konsol Amazon SQS melakukan panggilan `GetQueueAttributes` API saat halaman antrian dibuka. Permintaan `GetQueueAttributes` API mengaktifkan antrian.
> - Penundaan hingga 15 menit terjadi dalam CloudWatch metrik ketika antrian diaktifkan dari keadaan tidak aktif.
> - Tidak ada biaya untuk metrik Amazon SQS yang dilaporkan di. CloudWatch Mereka disediakan sebagai bagian dari layanan Amazon SQS.
> - CloudWatch metrik didukung untuk antrian standar dan FIFO.

Topik

Mengakses CloudWatch metrik untuk Amazon SQS

Amazon SQS dan Amazon CloudWatch terintegrasi sehingga Anda dapat menggunakan CloudWatch untuk melihat dan menganalisis metrik untuk antrian Amazon SQS Anda. Anda dapat melihat dan menganalisis metrik antrian dari konsol Amazon SQS, konsol, menggunakan, atau menggunakan API AWS CLI. CloudWatch CloudWatch Anda juga dapat mengatur CloudWatch alarm untuk metrik Amazon SQS.

Konsol Amazon SQS

1. Masuk ke konsol Amazon SQS.
2. Dalam daftar antrian, pilih (centang) kotak untuk antrian yang ingin Anda akses metrik. Anda dapat menampilkan metrik hingga 10 antrian.
3. Pilih tab Pemantauan.

   Berbagai grafik ditampilkan di bagian metrik SQS.
4. Untuk memahami apa yang diwakili grafik tertentu, arahkan kursor ke

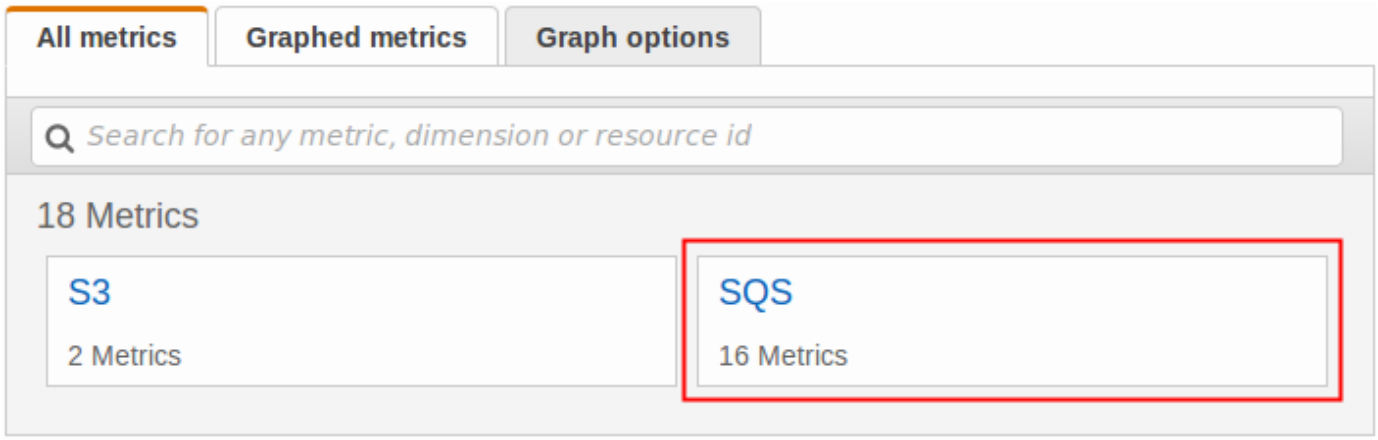
   

   samping grafik yang diinginkan, atau lihat CloudWatch Metrik yang tersedia untuk Amazon SQS.
5. Untuk mengubah rentang waktu untuk semua grafik secara bersamaan, untuk Rentang Waktu, pilih rentang waktu yang diinginkan (misalnya, Jam Terakhir).
6. Untuk melihat statistik tambahan untuk grafik individual, pilih grafik.
7. Di kotak dialog Detail CloudWatch Pemantauan, pilih Statistik, (misalnya, Jumlah). Untuk daftar statistik yang didukung, lihat CloudWatch Metrik yang tersedia untuk Amazon SQS.
8. Untuk mengubah rentang waktu dan interval waktu yang ditampilkan grafik individu (misalnya, untuk menampilkan rentang waktu 24 jam terakhir, bukan 5 menit terakhir, atau untuk menampilkan periode waktu setiap jam, bukan setiap 5 menit), dengan kotak dialog grafik masih ditampilkan, untuk Rentang Waktu, pilih rentang waktu yang diinginkan (misalnya, 24 Jam Terakhir). Untuk Periode, pilih periode waktu yang diinginkan dalam rentang waktu yang ditentukan (misalnya, 1 Jam). Setelah selesai melihat grafik, pilih Tutup.
9. (Opsional) Untuk bekerja dengan CloudWatch fitur tambahan, pada tab Pemantauan, pilih Lihat semua CloudWatch metrik, lalu ikuti petunjuk dalam CloudWatch Konsol Amazon prosedur.

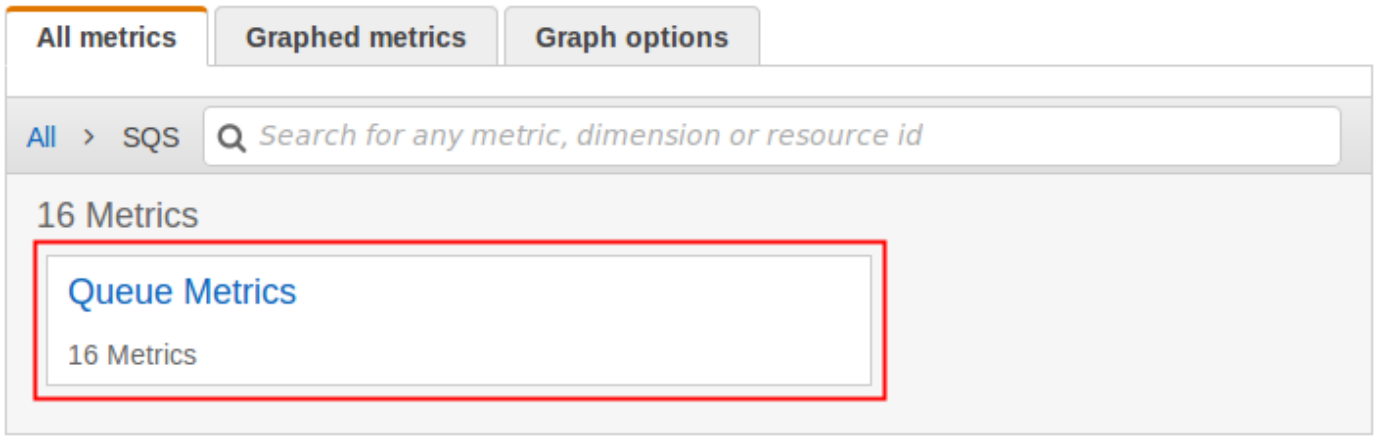
CloudWatch Konsol Amazon

1. Masuk ke konsol CloudWatch tersebut.

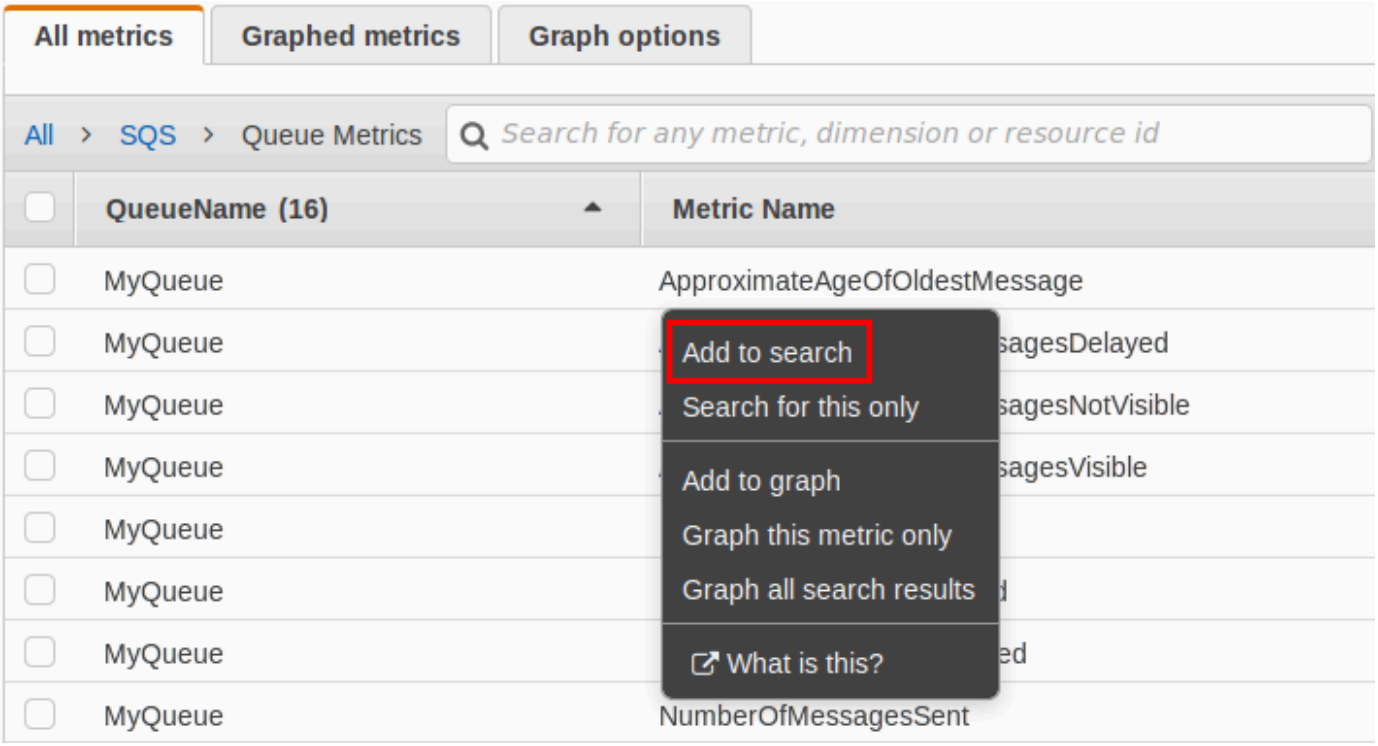
2. Di panel navigasi, pilih Metrics (Metrik).
3. Pilih namespace metrik SQS.



4. Pilih dimensi metrik Metrik Antrian.



5. Anda sekarang dapat memeriksa metrik Amazon SQS Anda:
   - Untuk mengurutkan metrik, gunakan judul kolom.
   - Untuk membuat grafik sebuah metrik, pilih kotak centang di samping metrik.
   - Untuk menyaring berdasarkan metrik, pilih nama metrik, kemudian pilih Tambahkan ke pencarian.



Untuk informasi selengkapnya dan opsi tambahan, lihat Metrik Grafik dan Menggunakan CloudWatch Dasbor Amazon di CloudWatch Panduan Pengguna Amazon.

AWS Command Line Interface

Untuk mengakses metrik Amazon SQS menggunakan AWS CLI, jalankan perintah. `get-metric-statistics`

Untuk informasi selengkapnya, lihat Mendapatkan Statistik untuk Metrik di Panduan CloudWatch Pengguna Amazon.

CloudWatch API

Untuk mengakses metrik Amazon SQS menggunakan CloudWatch API, gunakan tindakan. `GetMetricStatistics`

Untuk informasi selengkapnya, lihat Mendapatkan Statistik untuk Metrik di Panduan CloudWatch Pengguna Amazon.

Membuat CloudWatch alarm untuk metrik Amazon SQS

CloudWatch memungkinkan Anda memicu alarm berdasarkan ambang metrik. Misalnya, Anda dapat membuat alarm untuk `NumberOfMessagesSent` metrik. Misalnya, jika lebih dari 100 pesan dikirim ke `MyQueue` antrian dalam 1 jam, pemberitahuan email akan dikirim. Untuk informasi selengkapnya, lihat Membuat CloudWatch Alarm Amazon di Panduan CloudWatch Pengguna Amazon.

1. Masuk ke AWS Management Console dan buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Pilih Alarm, lalu pilih Buat Alarm.
3. Di bagian Pilih Metrik dari kotak dialog Buat Alarm, pilih Browse Metrics, SQS.
4. Untuk SQS > Metrik Antrian, pilih QueueNamedan Nama Metrik untuk mengatur alarm, lalu pilih Berikutnya. Untuk daftar metrik yang tersedia, lihat CloudWatch Metrik yang tersedia untuk Amazon SQS.

   Dalam contoh berikut, pemilihan adalah untuk alarm untuk `NumberOfMessagesSent` metrik untuk `MyQueue` antrian. Alarm terpicu ketika jumlah pesan yang dikirim melebihi 100.
5. Di bagian Tentukan Alarm pada kotak dialog Buat Alarm, lakukan hal berikut:

   a. Di bawah Ambang Alarm, ketik Nama dan Deskripsi untuk alarm.

   b. Set adalah ke > 100.

   c. Setel untuk 1 dari 1 titik data.

   d. Di bawah Pratinjau alarm, atur Periode ke 1 Jam.

   e. Atur Statistik ke Standar, Jumlah.

   f. Di bawah Tindakan, atur Setiap kali alarm ini ke Status adalah ALARM.

   Jika Anda CloudWatch ingin mengirim pemberitahuan saat alarm dipicu, pilih topik Amazon SNS yang ada atau pilih Daftar baru dan masukkan alamat email yang dipisahkan dengan koma.

   > **Note**
   >
   > Jika Anda membuat topik Amazon SNS baru, alamat email harus diverifikasi sebelum menerima pemberitahuan apa pun. Jika status alarm berubah sebelum alamat email diverifikasi, notifikasi tidak terkirim.
6. Pilih Buat Alarm.

Alarm dibuat.

CloudWatch Metrik yang tersedia untuk Amazon SQS

Amazon SQS mengirimkan metrik berikut ke. CloudWatch

> **Note**
>
> Untuk antrian standar, hasilnya adalah perkiraan karena arsitektur terdistribusi Amazon SQS. Dalam kebanyakan kasus, hitungan harus mendekati jumlah sebenarnya dari pesan dalam antrian.
> Untuk antrian FIFO, hasilnya tepat.

Metrik Amazon SQS

Namespace `AWS/SQS` mencakup metrik berikut.

Metrik	Deskripsi
`ApproximateAgeOfOldestMessage`	Perkiraan usia pesan tertua yang tidak dihapus dalam antrian. **Note** • Setelah pesan diterima tiga kali (atau lebih) dan tidak diproses, pesan dipindahkan ke bagian belakang antrian dan titik `ApproximateAgeOfOldestMessage` metrik pada pesan tertua kedua yang belum diterima lebih dari tiga kali. Tindakan ini terjadi bahkan jika antrian memiliki kebijakan redrive.

Metrik	Deskripsi
	• Karena satu pesan pil racun (diterima beberapa kali tetapi tidak pernah dihapus) dapat mendistorsi metrik ini, usia pesan pil racun tidak termasuk dalam metrik sampai pesan pil racun berhasil dikonsumsi. • Ketika antrian memiliki kebijakan redrive, pesan dipindahkan ke antrian huruf mati setelah jumlah maksimum penerima yang dikonfigurasi. Ketika pesan dipindahkan ke antrian huruf mati, `ApproximateAgeOfOldestMessage` metrik antrian huruf mati mewakili waktu ketika pesan dipindahkan ke antrian huruf mati (bukan waktu asli pesan dikirim). • Untuk antrian FIFO, pesan tidak dipindahkan ke bagian belakang antrian karena ini akan merusak jaminan pesanan FIFO. Pesan malah akan pergi ke DLQ jika ada yang dikonfigurasi. Jika tidak, itu akan memblokir grup pesan hingga berhasil dihapus atau sampai kedaluwarsa.

Metrik	Deskripsi
	Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif. Unit: detik Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)
`ApproximateNumberOfMessagesDelayed`	Jumlah pesan dalam antrian yang tertunda dan tidak tersedia untuk dibaca segera. Hal ini dapat terjadi ketika antrian dikonfigurasi sebagai antrian penundaan atau ketika pesan telah dikirim dengan parameter delay. Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif. Unit: Hitungan Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)

Metrik	Deskripsi
`ApproximateNumberOfMessagesNotVisible`	Jumlah pesan yang sedang dalam penerbangan. Pesan dianggap dalam penerbangan jika telah dikirim ke klien tetapi belum dihapus atau belum mencapai akhir jendela visibilitas mereka. Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif. Unit: Hitungan Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)
`ApproximateNumberOfMessagesVisible`	Jumlah pesan yang akan diproses. Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif. Unit: Hitungan Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS) Tidak ada batasan jumlah pesan untuk proses, namun Anda dapat membuat backlog ini menjadi periode retensi.

Metrik	Deskripsi
`NumberOfEmptyReceives` [1]	Jumlah panggilan `ReceiveMessage` API yang tidak mengembalikan pesan. Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif. Unit: Hitungan Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)

Metrik	Deskripsi
`NumberOfMessagesDeleted` [1]	Jumlah pesan yang dihapus dari antrian. Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif. Unit: Hitungan Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS) Amazon SQS memancarkan `NumberOfMessagesDeleted` metrik untuk setiap operasi penghapusan yang berhasil yang menggunakan pegangan tanda terima yang valid, termasuk penghapusan duplikat. Skenario berikut dapat menyebabkan nilai `NumberOfMessagesDeleted` metrik lebih tinggi dari yang diharapkan: • Memanggil `DeleteMessage` tindakan pada penanganan tanda terima yang berbeda yang termasuk dalam pesan yang sama: Jika pesan tidak diproses sebelum batas waktu visibilitas berakhir, pesan akan tersedia untuk konsumen lain yang dapat memprosesnya dan menghapusnya lagi, meningkatkan nilai metrik. `NumberOfMessagesDeleted` • Memanggil `DeleteMessage` tindakan pada pegangan tanda terima yang sama: Jika pesan diproses dan dihapus

Metrik	Deskripsi
	tetapi Anda memanggil `DeleteMessage` tindakan lagi menggunakan pegangan tanda terima yang sama, status keberhasilan akan dikembalikan, meningkatkan nilai `NumberOfMessagesDeleted` metrik.
`NumberOfMessagesReceived` [1]	Jumlah pesan yang dikembalikan oleh panggilan ke `ReceiveMessage` tindakan. Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif. Unit: Hitungan Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)

Metrik	Deskripsi
NumberOfMessagesSent [1]	Jumlah pesan yang ditambahkan ke antrian. Jika Anda mengirim pesan ke antrian huruf mati secara manual, pesan tersebut ditangkap oleh metrik. NumberOfMessagesSent Namun, jika pesan dikirim ke antrian huruf mati sebagai akibat dari upaya pemrosesan yang gagal, pesan tersebut tidak ditangkap oleh metrik ini. Dengan demikian, adalah mungkin untuk nilai-nilai NumberOfMessagesSent dan NumberOfMessagesReceived untuk menjadi berbeda. Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif. Unit: Hitungan Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)

Metrik	Deskripsi
`SentMessageSize` [1]	Ukuran pesan ditambahkan ke antrian. Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif. Unit: Bita Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS) **Note** `SentMessageSize` tidak ditampilkan sebagai metrik yang tersedia di CloudWatch konsol sampai setidaknya satu pesan dikirim ke antrian yang sesuai.

[1] Metrik ini dihitung dari perspektif layanan, dan dapat mencakup percobaan ulang. Jangan mengandalkan nilai absolut dari metrik ini, atau gunakan untuk memperkirakan status antrian saat ini.

Dimensi untuk metrik Amazon SQS

Satu-satunya dimensi yang dikirimkan Amazon SQS adalah. CloudWatch `QueueName` Ini berarti bahwa semua statistik yang tersedia disaring oleh`QueueName`.

Validasi kepatuhan untuk Amazon SQS

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat Program AWS Kepatuhan Program AWS .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Panduan Memulai Cepat Keamanan dan Kepatuhan — Panduan penerapan ini membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar AWS yang berfokus pada keamanan dan kepatuhan.
- Arsitektur untuk Keamanan dan Kepatuhan HIPAA di Amazon Web Services — Whitepaper ini menjelaskan bagaimana perusahaan dapat menggunakan AWS untuk membuat aplikasi yang memenuhi syarat HIPAA.

> **Note**
>
> Tidak semua memenuhi Layanan AWS syarat HIPAA. Untuk informasi selengkapnya, lihat Referensi Layanan yang Memenuhi Syarat HIPAA.

- AWS Sumber Daya AWS — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- AWS Panduan Kepatuhan Pelanggan - Memahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan untuk kontrol keamanan di berbagai kerangka kerja (termasuk Institut Standar dan Teknologi Nasional (NIST), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi (ISO)).
- Mengevaluasi Sumber Daya dengan Aturan dalam Panduan AWS Config Pengembang — AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- AWS Security Hub— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS. Security Hub menggunakan kontrol keamanan untuk sumber daya AWS Anda serta untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat Referensi kontrol Security Hub.
- Amazon GuardDuty — Ini Layanan AWS mendeteksi potensi ancaman terhadap beban kerja Akun AWS, kontainer, dan data Anda dengan memantau lingkungan Anda untuk aktivitas mencurigakan dan berbahaya. GuardDuty dapat membantu Anda mengatasi berbagai persyaratan kepatuhan, seperti PCI DSS, dengan memenuhi persyaratan deteksi intrusi yang diamanatkan oleh kerangka kerja kepatuhan tertentu.

- AWS Audit ManagerIni Layanan AWS membantu Anda terus mengaudit AWS penggunaan Anda untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap peraturan dan standar industri.

Ketahanan di Amazon SQS

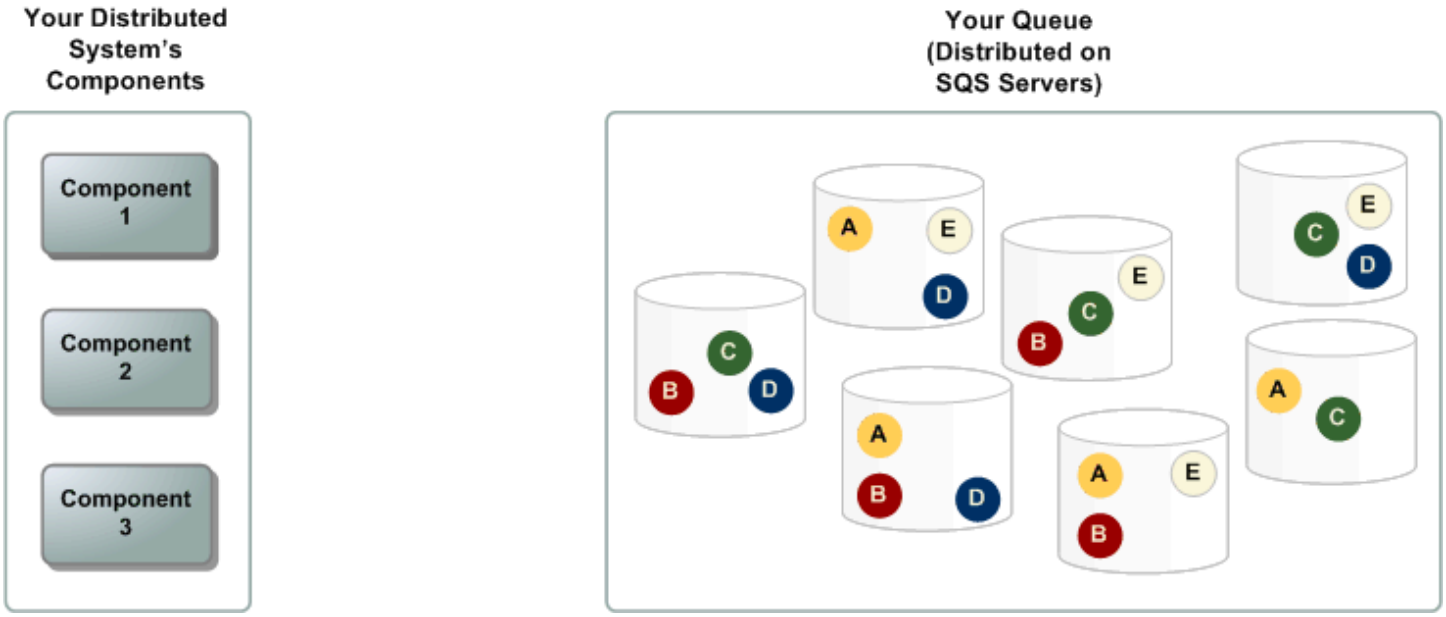
Infrastruktur AWS global dibangun di sekitar AWS Wilayah dan Zona Ketersediaan. AWS Wilayah menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan. Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan fail over di antara zona tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur pusat data tunggal atau multi tradisional. Untuk informasi selengkapnya tentang AWS Wilayah dan Availability Zone, lihat Infrastruktur AWS Global.

Selain infrastruktur AWS global, Amazon SQS menawarkan antrian terdistribusi.

Antrian terdistribusi

Ada tiga bagian utama dalam sistem pesan terdistribusi: komponen sistem terdistribusi Anda, antrian Anda (didistribusikan di server Amazon SQS), dan pesan dalam antrian.

Dalam skenario berikut, sistem Anda memiliki beberapa produsen (komponen yang mengirim pesan ke antrian) dan konsumen (komponen yang menerima pesan dari antrian). Antrian (yang menyimpan pesan A hingga E) menyimpan pesan secara berlebihan di beberapa server Amazon SQS.



Keamanan infrastruktur di Amazon SQS

Sebagai layanan terkelola, Amazon SQS dilindungi oleh prosedur keamanan jaringan AWS global yang dijelaskan dalam whitepaper Amazon Web Services: Overview of Security Processes.

Anda menggunakan tindakan API yang AWS dipublikasikan untuk mengakses Amazon SQS melalui jaringan. Klien harus mendukung Keamanan Lapisan Pengangkutan (TLS) 1.2 atau versi yang lebih baru. Selain itu, klien harus mendukung suite sandi dengan Perfect Forward Secrecy (PFS) seperti Ephemeral Diffie-Hellman (DHE) atau Elliptic Curve Diffie-Hellman Ephemeral (ECDHE).

Anda harus menandatangani permintaan menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan prinsipal IAM. Atau, Anda dapat menggunakanAWS Security Token Service (AWS STS) untuk menghasilkan kredensial keamanan sementara untuk permintaan penandatanganan.

Anda dapat memanggil tindakan API ini dari lokasi jaringan mana pun, tetapi Amazon SQS mendukung kebijakan akses berbasis sumber daya, yang dapat mencakup pembatasan berdasarkan alamat IP sumber. Anda juga dapat menggunakan kebijakan Amazon SQS untuk mengontrol akses dari titik akhir VPC Amazon tertentu atau VPC tertentu. Ini secara efektif mengisolasi akses jaringan ke antrian Amazon SQS tertentu dari hanya VPC tertentu dalam jaringan. AWS Untuk informasi selengkapnya, lihat Contoh 5: Tolak akses jika bukan dari titik akhir VPC.

Praktik terbaik keamanan Amazon SQS

AWS menyediakan banyak fitur keamanan untuk Amazon SQS, yang harus Anda tinjau dalam konteks kebijakan keamanan Anda sendiri. Berikut ini adalah praktik terbaik keamanan preventif untuk Amazon SQS.

> **Note**
>
> Panduan implementasi khusus yang diberikan adalah untuk kasus penggunaan umum dan implementasi. Kami menyarankan agar Anda melihat praktik terbaik ini dalam konteks kasus penggunaan, arsitektur, dan model ancaman spesifik Anda.

Topik

Pastikan antrian tidak dapat diakses publik

Kecuali Anda secara eksplisit meminta siapa pun di internet untuk dapat membaca atau menulis ke antrian Amazon SQS Anda, Anda harus memastikan bahwa antrian Anda tidak dapat diakses publik (dapat diakses oleh semua orang di dunia atau oleh pengguna yang diautentikasi). AWS

- Hindari pembuatan kebijakan dengan `Principal` diatur ke `""`.
- Hindari penggunaan wildcard (`*`). Sebagai gantinya, beri nama pengguna tertentu.

Menerapkan akses hak istimewa yang paling rendah

Saat Anda memberikan izin, Anda memutuskan siapa yang menerimanya, untuk antrian izin, dan tindakan API tertentu yang ingin Anda izinkan untuk antrian ini. Menerapkan hak istimewa paling sedikit penting untuk mengurangi risiko keamanan dan mengurangi efek kesalahan atau niat jahat.

Ikuti saran keamanan standar pemberian hak istimewa paling rendah. Artinya, hanya berikan izin yang diperlukan untuk melakukan tugas tertentu. Anda dapat menerapkan ini menggunakan kombinasi kebijakan keamanan.

Amazon SQS menggunakan model produsen-konsumen, yang membutuhkan tiga jenis akses akun pengguna:

- Administrator — Akses untuk membuat, memodifikasi, dan menghapus antrian. Administrator juga mengontrol kebijakan antrian.
- Produser — Akses untuk mengirim pesan ke antrian.
- Konsumen — Akses untuk menerima dan menghapus pesan dari antrian.

Untuk informasi selengkapnya, lihat bagian berikut:

- Manajemen identitas dan akses di Amazon SQS
- Izin Amazon SQS API: Tindakan dan referensi sumber daya
- Menggunakan kebijakan khusus dengan Bahasa Kebijakan Akses Amazon SQS

Gunakan peran IAM untuk aplikasi dan AWS layanan yang memerlukan akses Amazon SQS

Untuk aplikasi atau AWS layanan seperti Amazon EC2 untuk mengakses antrian Amazon SQS, mereka harus menggunakan kredensyal yang AWS valid dalam permintaan API mereka. AWS Karena kredensyal ini tidak diputar secara otomatis, Anda tidak boleh menyimpan AWS kredensyal secara langsung di aplikasi atau instans EC2.

Anda harus menggunakan peran IAM untuk mengelola kredensyal sementara untuk aplikasi atau layanan yang perlu mengakses Amazon SQS. Saat Anda menggunakan peran, Anda tidak perlu mendistribusikan kredensyal jangka panjang (seperti nama pengguna, kata sandi, dan kunci akses) ke instans atau AWS layanan EC2 seperti. AWS Lambda Sebagai gantinya, peran menyediakan izin sementara yang dapat digunakan aplikasi saat mereka melakukan panggilan ke AWS sumber daya lain.

Untuk informasi selengkapnya, lihat IAM Role dan Skenario Umum untuk Peran: Pengguna, Aplikasi, dan Layanan di Panduan Pengguna IAM.

Menerapkan enkripsi sisi server

Untuk mitigasi masalah kebocoran data, gunakan enkripsi saat istirahat untuk mengenkripsi pesan menggunakan kunci yang disimpan di lokasi berbeda dari lokasi yang menyimpan pesan Anda. Enkripsi sisi server (SSE) menyediakan enkripsi data saat istirahat. Amazon SQS mengenkripsi data Anda pada tingkat pesan saat menyimpannya, dan mendekripsi pesan untuk Anda saat Anda mengaksesnya. SSE menggunakan kunci yang dikelola di AWS Key Management Service. Selama Anda mengautentikasi permintaan Anda dan memiliki izin akses, tidak ada perbedaan antara mengakses antrian terenkripsi dan tidak terenkripsi.

Untuk informasi selengkapnya, lihat Enkripsi saat istirahat di Amazon SQS dan Manajemen Kunci Amazon SQS.

Menegakkan enkripsi data saat transit

Tanpa HTTPS (TLS), penyerang berbasis jaringan dapat menguping lalu lintas jaringan atau memanipulasinya, menggunakan serangan seperti. man-in-the-middle Izinkan hanya koneksi terenkripsi melalui HTTPS (TLS) menggunakan `aws:SecureTransport`kondisi dalam kebijakan antrian untuk memaksa permintaan menggunakan SSL.

Pertimbangkan untuk menggunakan titik akhir VPC untuk mengakses Amazon SQS

Jika Anda memiliki antrian yang harus dapat berinteraksi dengan Anda tetapi yang sama sekali tidak boleh diekspos ke internet, gunakan titik akhir VPC untuk mengantri akses hanya ke host dalam VPC tertentu. Anda dapat menggunakan kebijakan antrian untuk mengontrol akses ke antrian dari titik akhir VPC Amazon tertentu atau dari VPC tertentu.

Titik akhir Amazon SQS VPC menyediakan dua cara untuk mengontrol akses ke pesan Anda:

- Anda dapat mengontrol permintaan, pengguna, atau grup yang diizinkan melalui titik akhir VPC tertentu.
- Anda dapat mengontrol titik akhir VPC atau VPC mana yang memiliki akses ke antrian Anda menggunakan kebijakan antrian.

Lihat informasi yang lebih lengkap di Titik akhir Amazon Virtual Private Cloud untuk Amazon SQS dan Membuat kebijakan titik akhir Amazon VPC untuk Amazon SQS.

Sumber daya Amazon SQS terkait

Berikut ini adalah tabel yang mencantumkan daftar sumber daya terkait yang akan berguna saat Anda bekerja dengan layanan ini.

Sumber Daya	Deskripsi
Referensi API Layanan Antrian Sederhana Amazon	Deskripsi tindakan, parameter, dan tipe data dan daftar kesalahan yang dikembalikan layanan.
Amazon SQS di Referensi Perintah AWS CLI	Deskripsi AWS CLI perintah yang dapat Anda gunakan untuk bekerja dengan antrian.
Wilayah dan Titik Akhir	Informasi tentang wilayah dan titik akhir Amazon SQS
Halaman Produk	Halaman web utama untuk informasi tentang Amazon SQS.
Forum Diskusi	Forum berbasis komunitas bagi pengembang untuk mendiskusikan pertanyaan teknis yang terkait dengan Amazon SQS.
AWS Informasi Support Premium	Halaman web utama untuk informasi tentang Dukungan AWS Premium, saluran dukungan respons cepat satu-satu untuk membantu Anda membangun dan menjalank an aplikasi pada layanan infrastruktur. AWS

Riwayat dokumentasi

Tabel berikut menjelaskan perubahan penting pada Panduan Pengembang Layanan Antrian Sederhana Amazon sejak Januari 2019. Untuk pemberitahuan tentang pembaruan dokumentasi ini, berlangganan umpan RSS.

Fitur layanan terkadang diluncurkan secara bertahap ke AWS Wilayah tempat layanan tersedia. Kami memperbarui dokumentasi ini hanya untuk rilis pertama. Kami tidak memberikan informasi tentang ketersediaan Wilayah atau mengumumkan peluncuran Wilayah berikutnya. Untuk informasi tentang ketersediaan fitur layanan wilayah dan untuk berlangganan pemberitahuan tentang pembaruan, lihat Apa yang Baru dengan AWS? .

Perubahan	Deskripsi	Tanggal
AWS Protokol JSON	Buat permintaan API menggunakan protokol AWS JSON.	Juli 27, 2023
Bagian baru yang menjelaskan kebijakan AWS terkelola untuk Amazon SQS dan pembaruan kebijakan ini	Amazon SQS menambahk an tindakan baru yang memungkinkan Anda mencantumkan tugas pergerakan pesan terbaru (hingga 10) di bawah antrian sumber tertentu. Tindakan ini dikaitkan dengan operasi API `ListMessageMoveTasks` .	Juni 7, 2023
Penggerak ulang antrian huruf mati menggunakan API	Konfigurasikan redrive antrian huruf mati menggunakan Amazon SQS API.	Juni 7, 2023
ABAC untuk Amazon SQS	Kontrol akses berbasis atribut (ABAC) menggunakan tag antrian untuk izin akses yang fleksibel dan dapat diskalakan.	10 November 2022

Batas throughput tinggi FIFO meningkat	Peningkatan kuota default untuk mode throughput tinggi FIFO di Wilayah komersial, ditambah optimasi dokumen throughput tinggi FIFO.	20 Oktober 2022
Enkripsi sisi server default (SSE) tersedia	Enkripsi sisi server (SSE) menggunakan enkripsi milik SQS (SSE-SQS) secara default.	26 September 2022
Dukungan perlindungan wakil kebingungan Amazon SQS tersedia	Perlindungan wakil yang bingung memungkinkan Anda menentukan header baru dalam permintaan mereka, yang diperiksa terhadap kondisi dalam kebijakan KMS saat menggunakan SSE terkelola Amazon SQS.	Desember 29, 2021
SSE terkelola tersedia	Amazon SQS managed SSE (SSE-SQS) adalah enkripsi sisi server terkelola yang menggunakan kunci enkripsi milik Amazon SQS untuk melindungi data sensitif yang dikirim melalui antrian pesan.	23 November 2021
Penggerak ulang antrian huruf mati tersedia	Amazon SQS mendukung redrive antrian huruf mati untuk antrian standar.	November 10, 2021

Throughput tinggi untuk pesan dalam antrian FIFO tersedia	Throughput tinggi untuk antrian FIFO Amazon SQS memberikan jumlah transaksi per detik (TPS) yang lebih tinggi untuk pesan dalam antrian FIFO. Untuk informasi tentang kuota throughput, lihat Kuota yang terkait dengan pesan.	27 Mei 2021
Throughput tinggi untuk pesan dalam antrian FIFO tersedia dalam rilis pratinjau	Throughput tinggi untuk antrian Amazon SQS FIFO ada dalam rilis pratinjau dan dapat berubah sewaktu-waktu. Fitur ini memberikan jumlah transaksi per detik (TPS) yang lebih tinggi untuk pesan dalam antrian FIFO. Untuk informasi tentang kuota throughput, lihat Kuota yang terkait dengan pesan.	17 Desember 2020
Desain konsol Amazon SQS baru	Untuk menyederhanakan alur kerja pengembangan dan produksi, konsol Amazon SQS memiliki pengalaman pengguna baru.	8 Juli 2020
Amazon SQS mendukung pagination untuk ListQueues dan listDeadLetter SourceQueues	Anda dapat menentukan jumlah maksimum hasil yang akan dikembalikan dari ListQueues atau permintaan daftar. DeadLetter SourceQueues	22 Juni 2020

Amazon SQS mendukung metrik CloudWatch Amazon 1 menit di AWS semua Wilayah, kecuali AWS GovCloud Wilayah (AS)	CloudWatch Metrik satu menit untuk Amazon SQS tersedia di semua Wilayah, kecuali AWS GovCloud (US) Wilayah.	9 Januari 2020
Amazon SQS mendukung metrik 1 menit CloudWatch	CloudWatch Metrik satu menit untuk Amazon SQS saat ini hanya tersedia di Wilayah berikut: AS Timur (Ohio), Eropa (Irlandia), Eropa (Stockholm), dan Asia Pasifik (Tokyo).	25 November 2019
AWS Lambda pemicu untuk antrian Amazon SQS FIFO tersedia	Anda dapat mengonfigurasi pesan yang masuk dalam antrian FIFO sebagai pemicu fungsi Lambda.	25 November 2019
Enkripsi sisi server (SSE) untuk Amazon SQS tersedia di Wilayah China	SSE untuk Amazon SQS tersedia di Wilayah China.	13 November 2019
Antrian FIFO tersedia di Wilayah Timur Tengah (Bahrain)	Antrian FIFO tersedia di Wilayah Timur Tengah (Bahrain).	10 Oktober 2019
Titik akhir Amazon Virtual Private Cloud (Amazon VPC) untuk Amazon SQS tersedia di Wilayah AWS GovCloud (AS-Timur) dan (AS-Barat) AWS GovCloud	Anda dapat mengirim pesan ke antrian Amazon SQS dari Amazon VPC di AWS GovCloud Wilayah (AS-Timur) dan (AS-Barat). AWS GovCloud	Selasa, 05 September 2019

Amazon SQS memungkinkan pemecahan masalah antrian menggunakan atribut sistem pesan AWS X-Ray	Anda dapat memecahkan masalah pesan yang melewati antrian Amazon SQS menggunakan X-Ray. Rilis ini menambahkan parameter `MessageSystemAttribute` permintaan (yang memungkinkan Anda mengirim header jejak X-Ray melalui Amazon SQS) ke `SendMessage` operasi `SendMessageBatch` dan API, `AWSTraceHeader` atribut ke `ReceiveMessage` operasi API, dan tipe data`MessageSystemAttributeValue` .	28 Agustus 2019
Anda dapat menandai antrian Amazon SQS saat pembuatan	Anda dapat menggunakan satu panggilan Amazon SQS API, fungsi AWS SDK, atau perintah AWS Command Line Interface (AWS CLI) untuk secara bersamaan membuat antrian dan menentukan tag-nya. Selain itu, Amazon SQS mendukung kunci `aws:TagKeys` dan `aws:RequestTag` AWS Identity and Access Management (IAM).	22 Agustus 2019

Klien antrian sementara untuk Amazon SQS sekarang tersedia	Antrian sementara membantu Anda menghemat waktu pengembangan dan biaya penerapan saat menggunak an pola pesan umum seperti respon permintaan. Anda dapat menggunakan Klien Antrian Sementara untuk membuat antrean sementara yang dikelola aplikasi dengan throughput tinggi, hemat biaya, dan dikelola.	25 Juli 2019
SSE untuk Amazon SQS tersedia di Wilayah (AWS GovCloud AS-Timur)	Enkripsi sisi server (SSE) untuk Amazon SQS tersedia di Wilayah (AS-Timur). AWS GovCloud	20 Juni 2019
Antrian FIFO tersedia di Asia Pasifik (Hong Kong), China (Beijing), (AS-Timur), dan AWS GovCloud AWS GovCloud (AS-Barat)	Antrian FIFO tersedia di Asia Pasifik (Hong Kong), China (Beijing), (AS-Timur), dan AWS GovCloud AWS GovCloud (AS-Barat).	15 Mei 2019
Kebijakan titik akhir Amazon VPC tersedia untuk Amazon SQS	Anda dapat membuat kebijakan titik akhir Amazon VPC untuk Amazon SQS.	4 April 2019
Antrian FIFO tersedia di Wilayah Eropa (Stockholm) dan China (Ningxia)	Antrian FIFO tersedia di Wilayah Eropa (Stockholm) dan China (Ningxia).	14 Maret 2019

Antrian FIFO tersedia di semua Wilayah di mana Amazon SQS tersedia	Antrian FIFO tersedia di AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (California), AS Barat (Oregon), Asia Pasifik (Mumbai), Asia Pasifik (Seoul), Asia Pasifik (Singapura), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Kanada (Tengah), Eropa (Frankfurt), Eropa (Irlandia), Eropa (London), Eropa (Paris)), dan Wilayah Amerika Selatan (São Paulo).	7 Februari 2019

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.