



Panduan Developerr

Amazon Simple Queue Service



Amazon Simple Queue Service: Panduan Developer

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 Amazon SQS?	1
Manfaat menggunakan Amazon SQS	1
Arsitektur dasar	2
Antrean terdistribusi	2
Siklus hidup pesan	2
Perbedaan antara Amazon SQS, Amazon MQ, dan Amazon SNS	4
Menyiapkan	6
Langkah 1: Buat pengguna Akun AWS dan IAM	6
Mendaftar Akun AWS	6
Membuat pengguna administratif	7
Langkah 2: Berikan akses terprogram	8
Langkah 3: Bersiaplah untuk menggunakan kode contoh	10
Langkah selanjutnya	10
Memulai	11
Prasyarat	11
Memahami konsol Amazon SQS	11
Jenis antrian	12
Membuat antrian standar	13
Membuat antrean	13
Kirim pesan	16
Membuat antrian FIFO	16
Membuat antrean	16
Kirim pesan	19
Mengelola antrian	20
Prasyarat	11
Memahami konsol Amazon SQS	11
Mengedit antrian	21
Menerima dan menghapus pesan	22
Konfirmasikan antrian kosong	23
Hapus antrian	24
Bersihkan antrian	25
Tugas umum	26
Antrian standar	28
Pemesanan pesan	28

t-least-once Pengiriman	29
Antrian dan pengidentifikasi pesan	29
Pengidentifikasi untuk antrian standar	29
Kuota	30
Antrian FIFO	33
Logika pengiriman FIFO	34
Pemesanan pesan	35
Persis-sekali diproses	36
Pindah dari antrian standar ke antrian FIFO	36
Throughput tinggi untuk antrian FIFO	37
Partisi dan distribusi data	38
Aktifkan throughput tinggi untuk antrian FIFO	40
Istilah kunci	41
Kompatibilitas	42
Pengidentifikasi antrian dan pesan	43
Pengidentifikasi untuk antrian FIFO	29
Pengidentifikasi tambahan untuk antrian FIFO	44
Kuota	45
Kuota	47
Kuota yang terkait dengan pesan	47
Kuota terkait dengan kebijakan	51
Fitur dan kemampuan	53
Metadata pesan	53
Atribut pesan	53
Atribut sistem pesan	58
Sumber daya yang diperlukan untuk memproses pesan	58
Daftar paginasi	59
Tag alokasi biaya	60
Pemungutan suara pendek dan panjang	61
Mengkonsumsi pesan menggunakan pemungutan suara singkat	61
Mengkonsumsi pesan menggunakan polling panjang	62
Perbedaan antara pemungutan suara panjang dan pendek	63
Antrean surat mati	63
Bagaimana cara kerja antrean surat mati?	64
Apa manfaat antrian surat mati?	65
Bagaimana jenis antrian yang berbeda menangani kegagalan pesan?	66

Kapan saya harus menggunakan antrian huruf mati?	67
Memindahkan pesan dari antrian surat mati	68
Memecahkan masalah antrian surat mati	69
Mengonfigurasi antrean surat mati	70
Mengonfigurasi redrive antrian huruf mati	71
CloudTrail Persyaratan pembaruan dan izin	77
Batas waktu visibilitas	81
Dalam pesan penerbangan	83
Mengatur batas waktu visibilitas	84
Mengubah batas waktu visibilitas untuk pesan	85
Mengakhiri batas waktu visibilitas untuk pesan	85
Antrean tertunda	86
Antrian sementara	87
Antrian virtual	87
Pola pesan permintaan-respons (antrian virtual)	89
Contoh skenario: Memproses permintaan login	90
Membersihkan antrian	92
Pengatur waktu pesan	93
Mengakses pipa EventBridge	93
Mengelola pesan besar	95
Menggunakan Extended Client Library untuk Java	95
Menggunakan Extended Client Library untuk Python	105
Mengkonfigurasi Amazon SQS	109
ABAC untuk Amazon SQS	109
Apa itu ABAC?	109
Mengapa saya harus menggunakan ABAC di Amazon SQS?	110
Kunci kondisi ABAC untuk Amazon SQS	111
Penandaan untuk kontrol akses	111
Membuat pengguna IAM dan antrian Amazon SQS	112
Menguji kontrol akses berbasis atribut	116
Mengkonfigurasi parameter antrian	117
kebijakan kebijakan akses	119
Mengkonfigurasi SSE-SQS untuk antrian	120
Mengkonfigurasi SSE-KMS untuk antrian	121
Mengkonfigurasi tag untuk antrian	123
Berlangganan antrean ke topik	123

Mengkonfigurasi pemicu Lambda	124
Prasyarat	125
Atribut pesan	126
Praktik terbaik	128
Rekomendasi untuk antrian standar dan FIFO	128
Bekerja dengan pesan	128
Mengurangi biaya	132
Berpindah dari antrian Standar ke antrian FIFO	133
Rekomendasi tambahan untuk antrian FIFO	133
Menggunakan ID deduplikasi pesan	134
Menggunakan ID grup pesan	135
Menggunakan ID upaya permintaan terima	137
Contoh SDK Java	138
Menggunakan enkripsi sisi server	138
Menambahkan SSE ke antrian yang sudah ada	138
Menonaktifkan SSE untuk antrian	139
Membuat antrian dengan SSE	140
Mengambil atribut SSE	140
Mengonfigurasi tag	141
Mencantumkan tanda	141
Menambahkan atau memperbarui tag	141
Menghapus tanda	142
Atribut pesan	143
Mendefinisikan atribut	143
Mengirim pesan dengan atribut	145
Bekerja dengan JMS	146
Prasyarat	146
Memulai dengan Perpustakaan Java	148
Membuat koneksi JMS	148
Membuat antrian Amazon SQS	149
Mengirim pesan secara sinkron	150
Menerima pesan secara serentak	151
Menerima pesan secara asinkron	153
Menggunakan mode pengakuan klien	154
Menggunakan mode pengakuan unordered	155
Menggunakan Klien JMS dengan klien Amazon SQS lainnya	156

Contoh Java yang berfungsi untuk menggunakan JMS dengan antrian Standar Amazon SQS .	157
ExampleConfiguration.java	157
TextMessageSender.java	160
SyncMessageReceiver.java	162
AsyncMessageReceiver.java	164
SyncMessageReceiverClientAcknowledge.java	166
SyncMessageReceiverUnorderedAcknowledge.java	169
SpringExampleConfiguration.xml	173
SpringExample.java	174
ExampleCommon.java	177
Didukung implementasi JMS 1.1	178
Antarmuka umum yang didukung	178
Tipe pesan yang didukung	179
Mode pengakuan pesan yang didukung	179
JMS-didefinisikan header dan properti reserved	179
Tutorial	181
Membuat antrian Amazon SQS () AWS CloudFormation	181
Mengirim pesan dari VPC	183
Langkah 1: Buat pasangan kunci Amazon EC2	184
Langkah 2: Buat AWS sumber daya	184
Langkah 3: Konfirmasikan bahwa instans EC2 Anda tidak dapat diakses publik	185
Langkah 4: Buat titik akhir Amazon VPC untuk Amazon SQS	186
Langkah 5: Kirim pesan ke antrian Amazon SQS Anda	187
Mengotomatisasi dan pemecahan masalah	189
Mengotomatiskan notifikasi menggunakan EventBridge	189
Antrean pemecahan masalah menggunakan X-Ray	189
Keamanan	191
Perlindungan data	191
Enkripsi data	192
Privasi lalu lintas jaringan internet	204
Pengelolaan identitas dan akses	206
Audiens	206
Mengautentikasi dengan identitas	207
Mengelola akses menggunakan kebijakan	211
Ikhtisar	213
Bagaimana Amazon Simple Queue Service bekerja dengan IAM	221

Kebijakan terkelola AWS	229
Memecahkan masalah	230
Menggunakan kebijakan	232
Pencatatan dan pemantauan	280
Pencatatan panggilan API menggunakan CloudTrail	280
Memantau antrian menggunakan CloudWatch	298
Validasi kepatuhan	310
Ketahanan	312
Antrian terdistribusi	312
Keamanan infrastruktur	313
Praktik terbaik	313
Praktik terbaik pencegahan	313
Bekerja dengan API	317
Membuat permintaan API kueri menggunakan protokol AWS JSON	318
Membangun titik akhir	319
Membuat permintaan POST	320
Menafsirkan tanggapan Amazon SQS JSON API	320
FAQ protokol Amazon SQS AWS JSON	322
Membuat permintaan Query API dengan protokol AWS kueri	325
Membangun titik akhir	325
Membuat permintaan GET	326
Membuat permintaan POST	320
Menafsirkan respons API XML-AWS SQS	327
Mengautentikasi permintaan	329
Proses otentikasi dasar dengan HMAC-SHA	330
Bagian 1: Permintaan dari pengguna	331
Bagian 2: Tanggapan dari AWS	332
Batch	333
Mengaktifkan buffering sisi klien dan batching permintaan	334
Meningkatkan throughput menggunakan penskalaan horizontal dan tindakan batching	343
Sumber daya terkait	357
Riwayat dokumentasi	358
Glosarium AWS	365
.....	ccclxvi

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](#)
- [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 khusus 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 mengonsumsi 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.

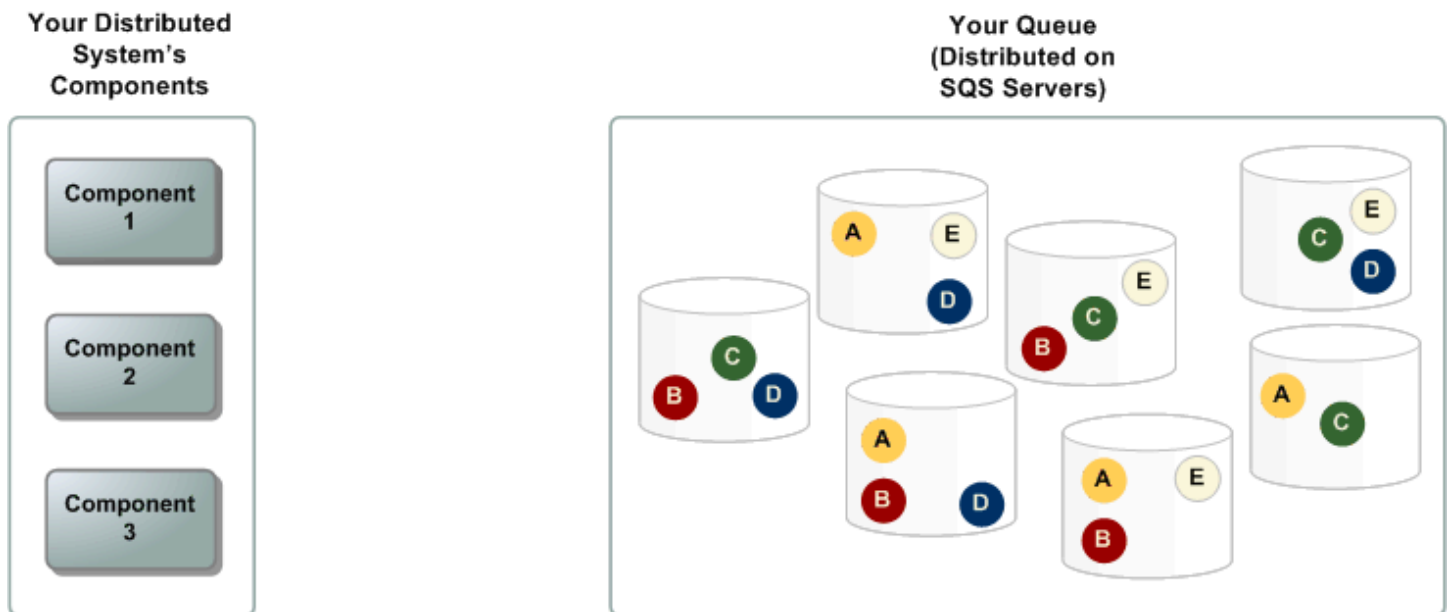
Arsitektur Amazon SQS

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

Antrean terdistribusi

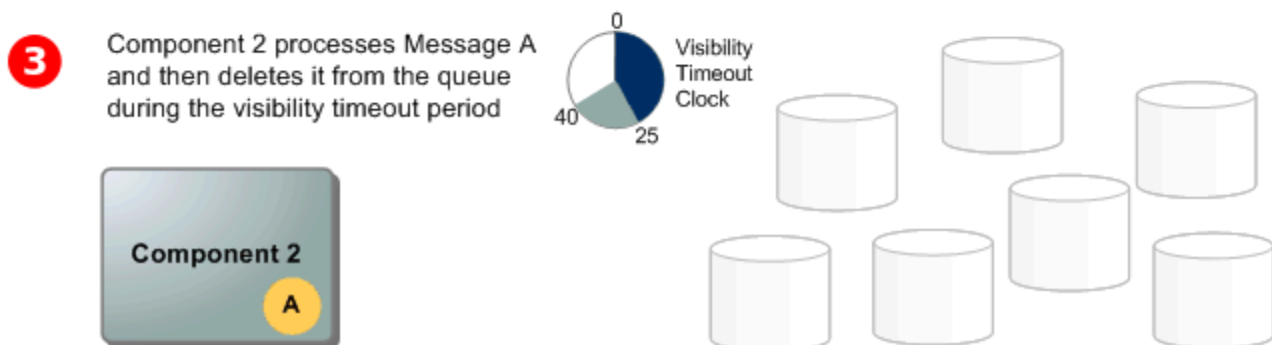
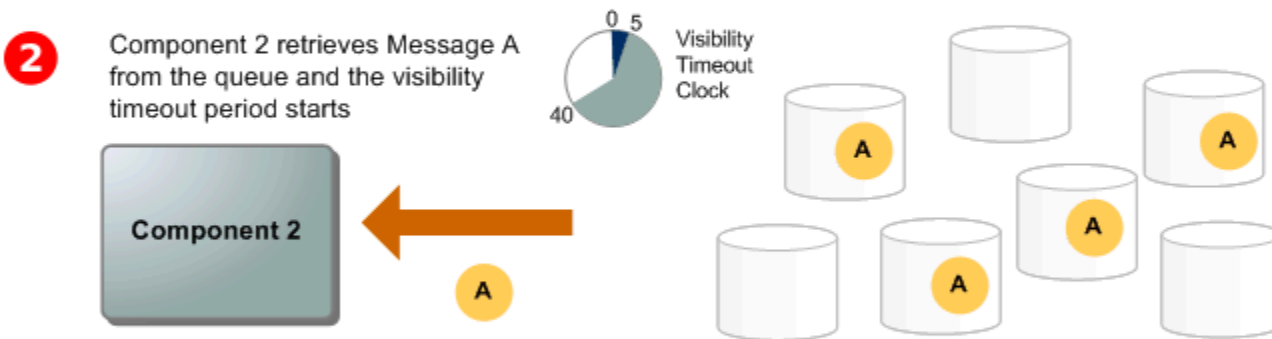
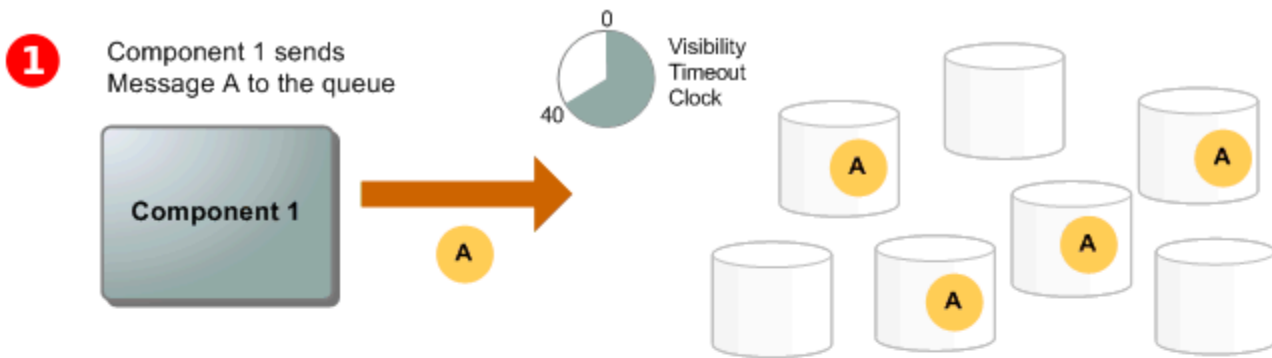
Ada tiga bagian utama dalam sistem pesan terdistribusi: komponen sistem terdistribusi Anda, antrian Anda (didistribusikan di server Amazon SQS), dan pesan dalam antrian.

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 melalui E) secara berlebihan menyimpan pesan di beberapa server Amazon SQS.



Siklus hidup pesan

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



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

Produk

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

Ketika

3

Konsumen

(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 dalam antrian selama lebih dari periode penyimpanan pesan. Periode retensi pesan default adalah 4 hari. Namun, Anda dapat mengatur periode retensi pesan ke nilai dari 60 detik ke 1.209.600 detik (14 hari) menggunakan [SetQueueAttributes](#) tindakan.

Perbedaan antara Amazon SQS, Amazon MQ, dan Amazon SNS

Amazon SQS, Amazon [SNS](#), dan [Amazon MQ](#) adalah layanan perpesanan terkelola yang sangat skalabel dan mudah digunakan. Berikut ini adalah ikhtisar perbedaan antara layanan ini:

Amazon SQS menawarkan antrian host yang mengintegrasikan dan memisahkan sistem dan komponen perangkat lunak terdistribusi. Amazon SQS menyediakan API layanan web generik yang dapat Anda akses menggunakan bahasa pemrograman apa pun yang didukung oleh SDK. AWS Pesan dalam antrian biasanya diproses oleh satu pelanggan. [Amazon SQS dan Amazon SNS sering digunakan bersama untuk membuat aplikasi pesan fanout.](#)

Amazon SNS adalah layanan berlangganan penerbitan yang menyediakan pengiriman pesan dari penerbit (juga dikenal sebagai produsen) ke beberapa titik akhir pelanggan (juga dikenal sebagai konsumen). Penerbit berkomunikasi secara asinkron dengan pelanggan dengan mengirim pesan ke topik, yang merupakan titik akses logis dan saluran komunikasi. Pelanggan dapat berlangganan topik Amazon SNS dan menerima pesan yang dipublikasikan menggunakan jenis titik akhir yang didukung, seperti, Amazon [Amazon Data Firehose](#), [SQS](#), [Lambda](#), HTTP, email, pemberitahuan push seluler, dan pesan teks seluler (SMS). Amazon SNS bertindak sebagai router pesan dan mengirimkan pesan ke pelanggan secara real time. Jika pelanggan tidak tersedia pada saat publikasi pesan, pesan tidak disimpan untuk pengambilan nanti.

Amazon MQ adalah layanan broker pesan terkelola yang menyediakan kompatibilitas dengan protokol pesan standar industri seperti Advanced Message Queueing Protocol (AMQP) dan Message Queuing Telemetry Transport (MQTT). [Amazon MQ saat ini mendukung jenis mesin Apache ActiveMQ dan RabbitMQ.](#)

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. 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](#)
- [Langkah 2: Berikan akses terprogram](#)
- [Langkah 3: Bersiaplah untuk menggunakan kode contoh](#)
- [Langkah selanjutnya](#)

Langkah 1: Buat pengguna Akun AWS dan IAM

Untuk mengakses AWS layanan apa pun, pertama-tama Anda harus membuat [Akun AWS](#), akun Amazon.com yang dapat menggunakan AWS produk. Anda dapat menggunakan Akun AWS untuk melihat laporan aktivitas dan penggunaan Anda serta untuk mengelola autentikasi dan akses.

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 Akun AWS

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

Untuk mendaftar Akun AWS

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

Anda akan diminta untuk menerima panggilan telepon dan memasukkan kode verifikasi pada keypad telepon sebagai bagian dari prosedur pendaftaran.

Saat Anda mendaftar Akun AWS, Pengguna root akun AWS akan dibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya dalam akun. Sebagai praktik terbaik keamanan, [tetapkan akses administratif ke pengguna administratif](#), dan hanya gunakan pengguna root untuk melakukan [tugas yang memerlukan akses pengguna root](#).

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

Membuat pengguna administratif

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

Mengamankan Pengguna root akun AWS Anda

1. Masuk ke [AWS Management Console](#) sebagai pemilik akun dengan memilih Pengguna root dan memasukkan alamat email Akun AWS Anda. Di halaman berikutnya, masukkan kata sandi Anda.

Untuk bantuan masuk menggunakan pengguna root, lihat [Masuk sebagai pengguna root](#) dalam Panduan Pengguna AWS Sign-In.

2. Aktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

Untuk petunjuknya, silakan lihat [Mengaktifkan perangkat MFA virtual untuk pengguna root Akun AWS Anda \(konsol\)](#) dalam Panduan Pengguna IAM.

Membuat pengguna 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 sebuah pengguna administratif.

Untuk mendapatkan tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, silakan lihat [Mengonfigurasi akses pengguna dengan Direktori Pusat Identitas IAM default](#) di Panduan Pengguna AWS IAM Identity Center.

Masuk sebagai pengguna administratif

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

Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat [Masuk ke portal akses AWS](#) dalam Panduan Pengguna AWS Sign-In.

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 ingin berinteraksi dengan AWS di luar AWS Management Console. Cara memberikan akses terprogram bergantung pada jenis pengguna yang mengakses AWS.

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

Pengguna mana yang membutuhkan akses terprogram?	Untuk	Oleh
Identitas tenaga kerja (Pengguna yang dikelola di Pusat Identitas IAM)	Gunakan kredensial sementara untuk menandatangani permintaan terprogram ke AWS CLI, SDK AWS, atau API AWS.	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. <ul style="list-style-type: none"> • Untuk AWS CLI, lihat Mengonfigurasi AWS CLI untuk menggunakan AWS IAM Identity Center di Panduan Pengguna AWS Command Line Interface. • Untuk SDK AWS, alat, dan API AWS, lihat Autentika

Pengguna mana yang membutuhkan akses terprogram?	Untuk	Oleh
		<p>si Pusat Identitas IAM di Panduan Referensi SDK dan Alat AWS.</p>
IAM	Gunakan kredensial sementara untuk menandatangani permintaan terprogram ke AWS CLI, SDK AWS, atau API AWS.	Mengikuti petunjuk dalam Menggunakan kredensial sementara dengan sumber daya AWS di Panduan Pengguna IAM.
IAM	(Tidak direkomendasikan) Gunakan kredensial jangka panjang untuk menandatangani permintaan terprogram ke AWS CLI, SDK AWS, atau API AWS.	<p>Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan.</p> <ul style="list-style-type: none"> • Untuk AWS CLI, lihat Mengautentikasi menggunakan kredensial pengguna IAM di Panduan Pengguna AWS Command Line Interface. • Untuk SDK dan alat AWS, lihat Mengautentikasi menggunakan kredensial jangka panjang di Panduan Referensi SDK dan Alat AWS. • Untuk API AWS, 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 persiapan 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 mengembangkan dan mengelola aplikasi di 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 dari Referensi [AWS Tools for PowerShellCmdlet](#).

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 belajar cara membuat antrian standar atau FIFO 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.

Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplication
MyTestQueue	Standard	6/27/2022, 13:03:08 EDT	0	0	Disabled	-
testFifo1.fifo	FIFO	6/27/2022, 13:03:41 EDT	0	0	Disabled	Disabled

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.

Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplication
MyTestQueue	Standard	6/27/2022, 13:03:08 EDT	0	0	Disabled	-
testFifo1.fifo	FIFO	6/27/2022, 13:03:41 EDT	0	0	Disabled	Disabled

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.

MyTestQueue

Edit Delete Purge Send and receive messages Start DLQ redrive

Details [Info](#)

Name MyTestQueue	Type Standard	ARN arn:aws:sqs:us-east-1:269704527654:MyTestQueue
Encryption Disabled	URL https://sqs.us-east-1.amazonaws.com/269704527654/MyTestQueue	Dead-letter queue -



► More

SNS subscriptions Lambda triggers Dead-letter queue Monitoring Tagging Access policy Encryption Dead-letter queue redrive tasks

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 antrian standar Amazon SQS](#) [Memulai antrian Amazon SQS FIFO](#)

Antrian standar	Antrian FIFO
<p>Throughput Tanpa Batas — Antrian standar mendukung jumlah panggilan API yang hampir tidak terbatas per detik, per tindakan API (SendMessage ,ReceiveMessage , atau DeleteMessage</p>	<p>Throughput Tinggi — Jika Anda menggunakan an batching, antrian FIFO mendukung hingga 3.000 pesan per detik, per metode API (SendMessageBatch ,, atau ReceiveMessage DeleteMessageBatch 3.000 pesan per detik mewakili 300 panggilan API, masing-masing dengan batch 10 pesan. Untuk meminta peningkatan kuota, kiriman permintaan dukungan. Tanpa batching, antrian FIFO mendukung hingga 300 panggilan API per detik, per metode API (SendMessage ,, atau ReceiveMessage DeleteMessage</p>
<p>Pengiriman At-Least-Once — Pesan dikirim setidaknya sekali, tetapi kadang-kadang lebih dari satu salinan pesan terkirim.</p>	<p>Exactly-Once Processing — Pesan dikirimkan sekali dan tetap tersedia sampai konsumen</p>
<p>Pemesanan Upaya Terbaik - Kadang-kadang, pesan dikirim dalam urutan yang berbeda dari yang dikirim.</p>	

Antrian standar	Antrian FIFO
	<p>memproses dan menghapusnya. Duplikat tidak dimasukkan ke dalam antrian.</p> <p>Pengiriman First-In-First-Out - Urutan pengiriman dan penerimaan pesan dipertahankan dengan ketat.</p> 
<p>Kirim data antar aplikasi saat throughput penting, misalnya:</p> <ul style="list-style-type: none"> • 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. 	<p>Kirim data antar aplikasi saat urutan acara penting, misalnya:</p> <ul style="list-style-type: none"> • Pastikan perintah yang dimasukkan pengguna dijalankan dalam urutan yang benar. • Tampilkan harga produk yang benar dengan mengirimkan modifikasi harga dalam urutan yang benar. • 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 (konsol)

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 antrian standar Amazon SQS

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

ℹ Note

Anda tidak dapat mengubah jenis antrian setelah 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 default adalah 4 hari.
 - c. Untuk keterlambatan Pengiriman, masukkan durasi dan unit. Kisarannya dari 0 detik hingga 15 menit. Nilai default adalah 0 detik.
 - d. Untuk Ukuran pesan maksimum, masukkan nilai. Kisarannya dari 1 KB hingga 256 KB. Nilai default adalah 256 KB.
 - e. Untuk Menerima waktu tunggu pesan, masukkan nilai. Kisarannya dari 0 hingga 20 detik. Nilai default 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 \(SSE\) untuk antrian menggunakan kunci enkripsi yang dikelola SQS \(konsol\)](#) dan [Mengkonfigurasi enkripsi sisi server \(SSE\) untuk antrian \(konsol\)](#).

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 antrian

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/](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. Lihat informasi yang lebih lengkap di [Throughput tinggi untuk antrian FIFO](#) dan [Kuota yang terkait dengan pesan](#).

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 Remove, 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 \(SSE\) untuk antrian menggunakan kunci enkripsi yang dikelola SQS \(konsol\)](#) dan [Mengkonfigurasi enkripsi sisi server \(SSE\) untuk antrian \(konsol\)](#).

 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 antrian. 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 antrian First-In-First-Out (FIFO), masukkan ID grup Pesan. Untuk informasi selengkapnya, lihat [Logika pengiriman FIFO](#).
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 FIFO](#).
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

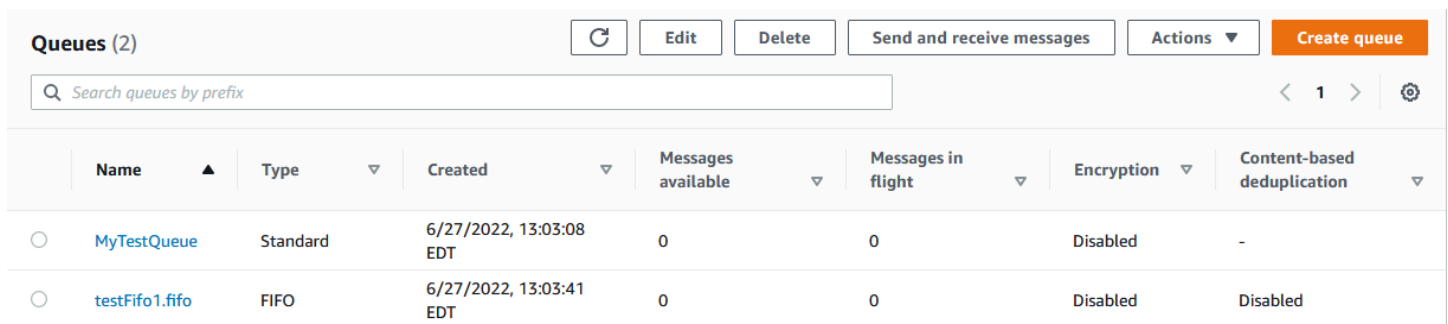
Bagian ini membantu Anda menjadi lebih akrab dengan Amazon SQS dengan menunjukkan cara mengelola antrian dan pesan menggunakan konsol Amazon SQS.

Prasyarat

Sebelum memulai, selesaikan langkah-langkah di [Menyiapkan Amazon SQS](#).

Memahami konsol Amazon SQS

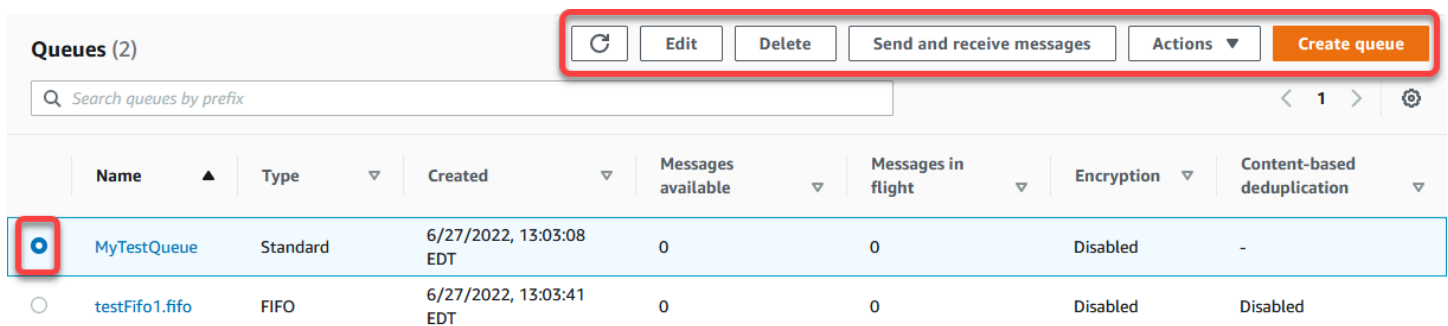
Saat Anda membuka konsol, pilih Antrian dari panel navigasi untuk menampilkan halaman Antrian. Halaman Antrian memberikan informasi tentang semua antrian Anda di wilayah aktif.



Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplication
MyTestQueue	Standard	6/27/2022, 13:03:08 EDT	0	0	Disabled	-
testFifo1.fifo	FIFO	6/27/2022, 13:03:41 EDT	0	0	Disabled	Disabled

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.



Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplication
MyTestQueue	Standard	6/27/2022, 13:03:08 EDT	0	0	Disabled	-
testFifo1.fifo	FIFO	6/27/2022, 13:03:41 EDT	0	0	Disabled	Disabled

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.

MyTestQueue

Edit Delete Purge Send and receive messages Start DLQ redrive

Details info

Name	Type	ARN
MyTestQueue	Standard	arn:aws:sqs:us-east-1:269704527654:MyTestQueue
Encryption	URL	Dead-letter queue
Disabled	https://sqs.us-east-1.amazonaws.com/269704527654/MyTestQueue	-

► More

SNS subscriptions Lambda triggers Dead-letter queue Monitoring Tagging Access policy Encryption Dead-letter queue redrive tasks

Mengedit antrian (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 antrian [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 (konsol)

Setelah mengirim pesan ke antrian, Anda dapat menerima dan menghapusnya. Saat Anda meminta pesan dari antrian, Anda tidak dapat menentukan pesan mana yang akan diambil. Sebagai gantinya, Anda menentukan jumlah maksimum pesan (hingga 10) yang ingin Anda ambil.

Note

Karena Amazon SQS adalah sistem terdistribusi, antrian dengan sangat sedikit pesan mungkin menampilkan respons kosong untuk permintaan terima. Dalam hal ini, jalankan kembali permintaan untuk mendapatkan pesan Anda. Tergantung pada kebutuhan aplikasi Anda, Anda mungkin harus menggunakan [polling pendek atau panjang](#) untuk menerima pesan.

Amazon SQS tidak secara otomatis menghapus pesan setelah mengambilnya untuk Anda, jika Anda tidak berhasil menerima pesan (misalnya, jika konsumen gagal atau Anda kehilangan konektivitas). Untuk menghapus pesan, Anda harus mengirim permintaan terpisah yang mengakui bahwa Anda telah berhasil menerima dan memproses pesan. Perhatikan bahwa Anda harus menerima pesan sebelum Anda dapat menghapusnya.

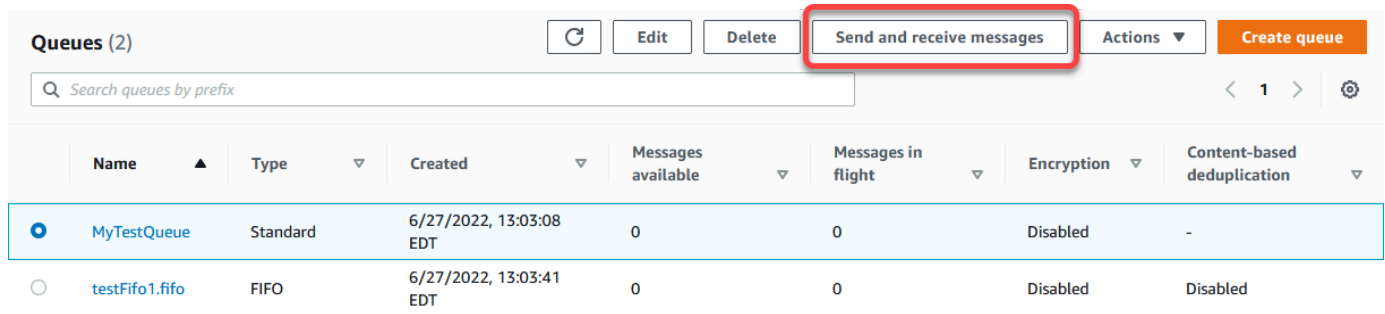
Note

Setelah menerima pesan dari konsol Amazon SQS, konsol segera menyetel pesan kembali ke terlihat, sehingga pesan dapat diterima lagi.

Untuk informasi selengkapnya tentang opsi API untuk menerima dan menghapus pesan, lihat Panduan Referensi API Amazon SQS.

Untuk menerima dan menghapus pesan (konsol)

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



The screenshot shows the Amazon SQS console interface. At the top, there are buttons for 'Send and receive messages' (highlighted with a red box), 'Actions', and 'Create queue'. Below the buttons is a search bar and a table of queues. The table has columns for Name, Type, Created, Messages available, Messages in flight, Encryption, and Content-based deduplication. Two queues are listed: 'MyTestQueue' (Standard) and 'testFifo1.fifo' (FIFO).

Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplication
MyTestQueue	Standard	6/27/2022, 13:03:08 EDT	0	0	Disabled	-
testFifo1.fifo	FIFO	6/27/2022, 13:03:41 EDT	0	0	Disabled	Disabled

Konsol menampilkan halaman Kirim dan terima pesan.

5. 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.

6. Untuk menghapus pesan, pilih pesan yang ingin dihapus dan pilih Hapus.
7. Di kotak dialog Hapus Pesan, pilih Hapus.

Mengonfirmasi bahwa antrian 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 antrian masih berisi pesan, terutama jika Anda menetapkan nilai rendah untuk Menerima pesan waktu tunggu saat Anda membuat antrian. 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/](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](#).

Hapus antrian

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

 Tip

Jika Anda ingin memverifikasi bahwa antrian kosong sebelum Anda menghapusnya, lihat [Mengonfirmasi bahwa antrian 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/](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, konfirmasi penghapusan dengan memasukkan **delete**.
6. Pilih Hapus.


Untuk menghapus antrian (AWS CLI/AWS API)

Anda dapat menggunakan salah satu perintah berikut untuk menghapus antrian:

- AWS CLI: [aws sqs delete-queue](#)
- AWS API: [DeleteQueue](#)

Membersihkan pesan dari antrean Amazon SQS

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

 Important

Saat Anda membersihkan antrean, 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 Tindakan, pilih Purge.
5. Di kotak dialog Purge queue, konfirmasi pembersihan dengan memasukkan **purge** dan memilih Purge.

Semua pesan dibersihkan 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. [Mengkonfigurasi antrian untuk memicu AWS Lambda fungsi \(konsol\)](#)
- 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](#)
- 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 [Kode Sampel dan Pustaka serta pusat](#) pengembang:
 - [Java](#)
 - [JavaScript](#)

- [PHP](#)
 - [Python](#)
 - [Ruby](#)
 - [Windows & .NET](#)
-
- Pelajari tentang mengawasi biaya dan sumber daya di [Mengotomatisasi dan pemecahan masalah antrean Amazon SQS](#) bagian ini.
 - Pelajari cara melindungi data Anda dan mengaksesnya di bagian [Keamanan](#).
 - Pelajari lebih lanjut tentang alur kerja dan proses Amazon SQS:

Memulai 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 \(konsol\)](#) Untuk contoh Java, lihat [Contoh 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 antrean 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 tersebut, 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.

Topik

- [Pengidentifikasi untuk antrian Standar Amazon SQS](#)

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+JvwvJaBV+3dcjk2YW2vA3+STFF1jTM8tJJg6HRG6PYSasuWXPJB+Cw
Lj1FjgXUv1uSj1gUPAWV66FU/WeR4mq20KpEGYWbnLmpRCJVAyeMjeU5ZBdteQ+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
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)	<p>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 <code>OverLimit</code> 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.</p>
Nama antrian	<p>Nama antrian dapat memiliki hingga 80 karakter. Karakter berikut diterima: karakter alfanumerik, tanda hubung (-), dan garis bawah (_).</p> <div data-bbox="688 1591 1507 1864"><p> Note</p><p>Nama antrian peka huruf besar/kecil (misalnya, <code>Test-queue</code> dan antrian <code>test-queue</code> yang berbeda).</p></div>

Kuota	Deskripsi
Tag antrian	<p>Kami tidak menyarankan menambahkan lebih dari 50 tag ke antrian. Penandaan mendukung karakter Unicode di UTF-8.</p> <p>Tag Key diperlukan, tetapi tag Value adalah opsional.</p> <p>Tag Key dan tag Value peka huruf besar/kecil.</p> <p>Tag Key dan tag Value dapat menyertakan karakter alfanumerik Unicode di UTF-8 dan spasi putih. Karakter khusus berikut diperbolehkan: <code>_ . : / = + - @</code></p> <p>Tag Key atau tidak Value boleh menyertakan awalan cadangan aws : (Anda tidak dapat menghapus kunci tag atau nilai dengan awalan ini).</p> <p>KeyPanjang tag maksimum adalah 128 karakter Unicode di UTF-8. Tag tidak Key boleh kosong atau null.</p> <p>ValuePanjang tag maksimum adalah 256 karakter Unicode di UTF-8. Tag Value mungkin kosong atau nol.</p> <p>Tindakan penandaan dibatasi hingga 30 TPS per. Akun AWS Jika aplikasi Anda membutuhkan throughput yang lebih tinggi, kirimkan permintaan.</p>

Memulai antrian Amazon SQS FIFO

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:

- Sistem manajemen pesanan e-Commerce di mana pesanan sangat penting
- Mengintegrasikan dengan sistem pihak ketiga di mana acara perlu diproses secara berurutan
- Memproses input yang dimasukkan pengguna dalam urutan yang dimasukkan
- Komunikasi dan jaringan — Mengirim dan menerima data dan informasi dalam urutan yang sama
- Sistem komputer — Memastikan bahwa perintah yang dimasukkan pengguna dijalankan dalam urutan yang benar
- Lembaga pendidikan — Mencegah siswa mendaftar di kursus sebelum mendaftar untuk akun
- 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, lihat [Throughput tinggi untuk antrian FIFO](#). Untuk informasi tentang kuota throughput, lihat [the section called “Kuota yang terkait dengan pesan”](#)

Antrian Amazon SQS FIFO tersedia di semua Wilayah di mana 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 \(konsol\)](#) Untuk contoh Java, lihat [Contoh Amazon SQS Java SDK](#).

Untuk praktik terbaik bekerja dengan antrian FIFO, lihat dan [Rekomendasi tambahan untuk antrian FIFO Amazon SQS](#) [Rekomendasi untuk standar Amazon SQS dan antrian FIFO](#)

Logika pengiriman FIFO

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 antrian 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 antrian 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 produsen 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.

Pemesanan pesan

[Antrian FIFO meningkatkan dan melengkapinya 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 diurutkan dalam satu antrian. Tidak ada kuota untuk jumlah grup pesan dalam antrian FIFO.

Persis-sekali diproses

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

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 antrian 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 pemesanan, lihat [Kuota yang terkait dengan pesan](#).

- 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, konfirmasi 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

Throughput tinggi untuk [antrian FIFO](#) mendukung jumlah permintaan per API yang lebih tinggi, per detik. Untuk meningkatkan jumlah permintaan dalam throughput tinggi untuk antrian FIFO, Anda dapat meningkatkan jumlah grup pesan yang Anda gunakan. Untuk informasi selengkapnya tentang kuota pesan throughput tinggi, lihat kuota [layanan Amazon SQS](#) di kuota. Referensi Umum Amazon Web Services Untuk informasi tentang kuota per antrian dengan throughput tinggi untuk kuota FIFO, lihat dan. [Kuota yang terkait dengan pesan Partisi dan distribusi data untuk throughput tinggi untuk antrian SQS FIFO](#)

Topik

- [Partisi dan distribusi data untuk throughput tinggi untuk antrian SQS FIFO](#)
- [Aktifkan throughput tinggi untuk antrian FIFO](#)

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:

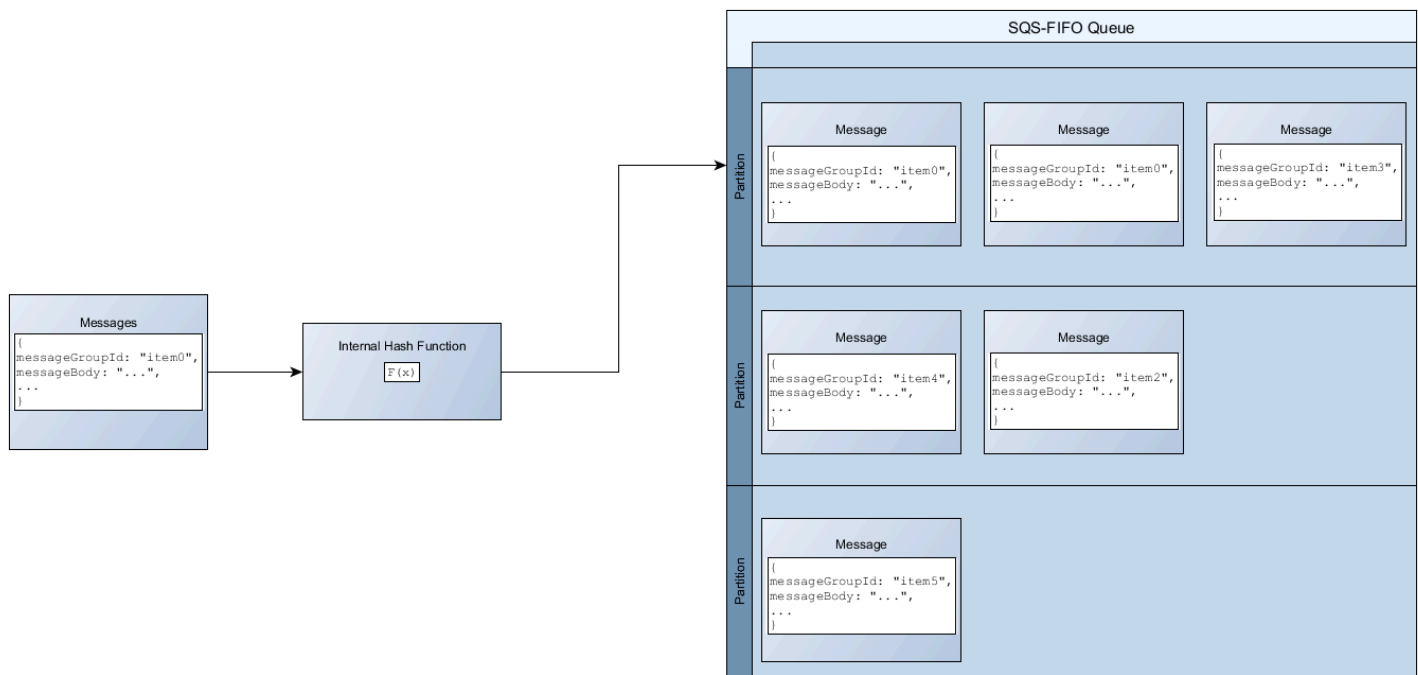
- Jika tingkat permintaan saat ini mendekati atau melebihi apa yang dapat didukung oleh partisi yang ada, partisi tambahan dialokasikan hingga antrian mencapai kuota regional. Untuk informasi tentang kuota, lihat [Kuota yang terkait dengan pesan](#).
- Jika partisi saat ini memiliki pemanfaatan rendah, 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

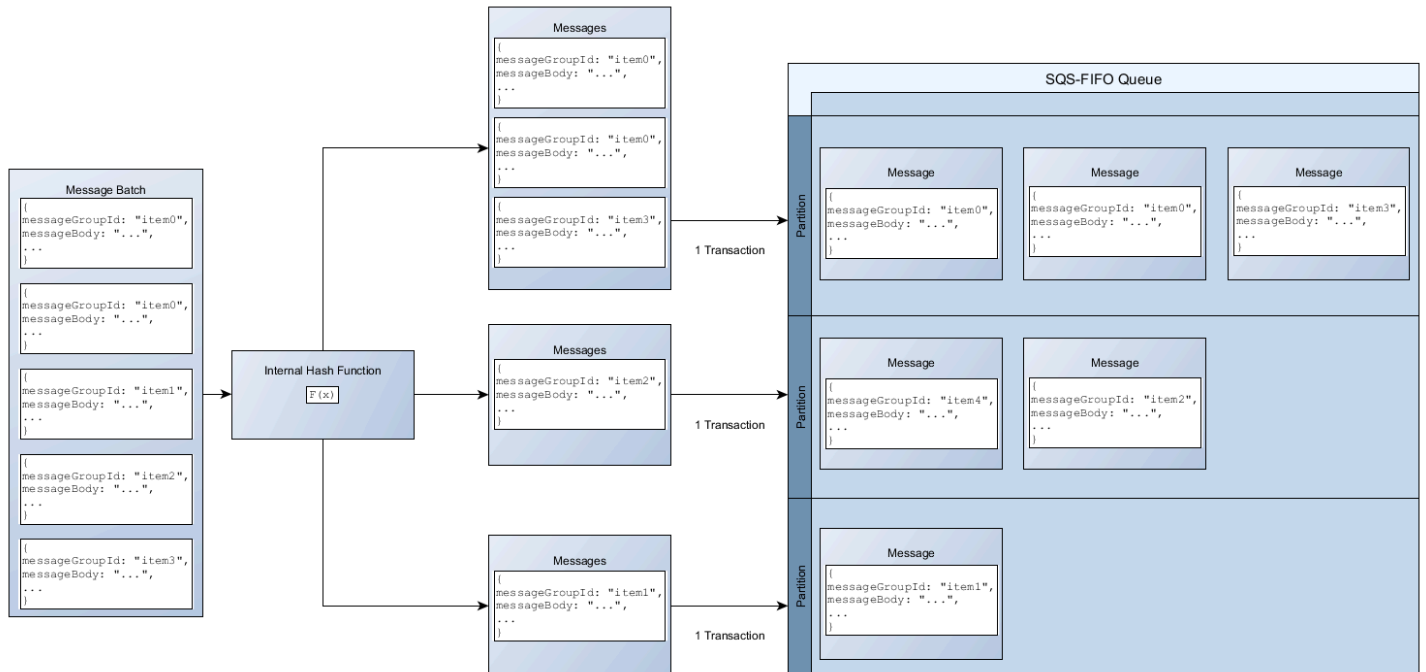
Saat menggunakan API batch, setiap pesan dirutekan berdasarkan proses yang dijelaskan dalam [Mendistribusikan data dengan ID grup pesan](#). Pesan yang dialihkan ke partisi yang sama dikelompokkan dan diproses dalam satu transaksi.

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 merekomendasikan 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

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 yang terkait dengan pesan](#)

Istilah kunci

Istilah-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

Klien

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

Layanan

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 Amazon SQS dan pengidentifikasi pesan

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

Topik

- [Pengidentifikasi untuk antrian Amazon SQS FIFO](#)
- [Pengidentifikasi tambahan untuk antrian Amazon SQS FIFO](#)

Pengidentifikasi untuk antrian Amazon SQS FIFO

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+STFF1jTM8tJJg6HRG6PYSasuWXPJB+Cw
Lj1FjgXUv1uSj1gUPAWV66FU/WeR4mq20KpEGYWbnLmpRCJVAyeMjeU5ZBdteQ+QE
auMZc8ZRv37sIW2iJKq3M9MFx1YvV11A2x/KSbkJ0=
```

Pengidentifikasi tambahan untuk antrian Amazon SQS FIFO

Untuk informasi selengkapnya tentang pengidentifikasi berikut, lihat [Persis-sekali diproses](#) dan 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

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.
Nama antrian	Nama antrian FIFO harus diakhiri dengan akhiran. <code>.fifo</code> Akhiran dihitung terhadap kuota nama antrian 80

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

Kuota Amazon SQS

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

Topik

- [Kuota yang terkait dengan pesan](#)
- [Kuota terkait dengan kebijakan](#)

Kuota yang terkait dengan pesan


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 Mengonfigurasi AmazonSQSBufferedAsyncClient di bagian Tindakan batch Amazon SQS .
Konten pesan	<p>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</p> <p>Karakter apa pun yang tidak termasuk dalam daftar ini ditolak. Untuk informasi lebih lanjut, lihat spesifikasi W3C untuk karakter.</p>
ID grup pesan	Gunakan pesan dari backlog untuk menghindari membangun backlog besar pesan dengan ID grup pesan yang sama .

Kuota	Deskripsi
	<p>MessageGroupId diperlukan untuk antrian FIFO. Anda tidak dapat menggunakannya untuk antrian Standar.</p> <p>Anda harus mengaitkan yang tidak kosong MessageGroupId dengan pesan. Jika Anda tidak memberikan MessageGroupId, tindakan gagal.</p> <p>MessageGroupId Panjangnya 128 karakter. Nilai yang valid: karakter alfanumerik dan tanda baca. (!"#\$%&'()*+, -./:;<=>?@[\\]^_`{ }~)</p>
Retensi pesan	<p>Secara default, pesan disimpan selama 4 hari. Minimal adalah 60 detik (1 menit). Maksimum adalah 1.209.600 detik (14 hari).</p>
Throughput pesan	<p>Antrian standar mendukung jumlah panggilan API yang hampir tidak terbatas per detik, per tindakan API (SendMessage, ReceiveMessage, atau DeleteMessage).</p> <p>Antrian FIFO</p> <ul style="list-style-type: none"> • 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. Untuk meminta peningkatan kuota, kirimkan permintaan dukungan.

Kuota	Deskripsi
	<p data-bbox="686 226 1211 262"><u>Throughput tinggi untuk antrian FIFO</u></p> <ul data-bbox="686 310 1502 1533" style="list-style-type: none"><li data-bbox="686 310 1502 535">• Tanpa batching (<code>SendMessage</code> , <code>ReceiveMessage</code> , dan <code>DeleteMessage</code>), 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).<li data-bbox="686 556 1502 682">• Untuk Wilayah AS Timur (Ohio) dan Eropa (Frankfurt), throughput default adalah 18.000 transaksi per detik per aksi API.<li data-bbox="686 703 1502 882">• 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.<li data-bbox="686 903 1502 1039">• Untuk Eropa (London) dan Amerika Selatan (São Paulo), throughput default adalah 4.500 transaksi per detik per aksi API.<li data-bbox="686 1060 1502 1186">• Untuk throughput maksimum, tingkatkan jumlah ID grup pesan yang Anda gunakan untuk pesan yang dikirim tanpa pengelompokan.<li data-bbox="686 1207 1502 1533">• Anda dapat meningkatkan throughput hingga 700.000 pesan per detik dengan menggunakan API batching (<code>SendMessageBatch</code> dan <code>DeleteMessageBatch</code>) 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. <p data-bbox="716 1575 1485 1795">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.</p>

Kuota	Deskripsi
	<p>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 <code>SendMessageBatch</code> dan <code>DeleteMessageBatch</code> , semua pesan dalam permintaan batch harus menggunakan ID grup pesan yang sama.</p> <ul style="list-style-type: none">• 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 <code>SendMessageBatch</code> dan <code>DeleteMessageBatch</code> , 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 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	<p>Ukuran pesan minimum adalah 1 byte (1 karakter). Maksimum adalah 262.144 byte (256 KiB).</p> <p>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.</p> <div data-bbox="685 716 1507 936" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> Note</p> <p>Pustaka yang diperluas ini hanya berfungsi untuk klien sinkron.</p> </div>
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. Lihat informasi yang lebih lengkap di Kuota terkait dengan kebijakan .

Kuota terkait dengan kebijakan

Tabel berikut mencantumkan kuota yang terkait dengan kebijakan.

Nama	Maksimum
Byte	8.192
Kondisi	10
Penanggung jawab	50

Nama	Maksimum
Pernyataan	20
Tindakan per pernyataan	7

Fitur dan kemampuan Amazon SQS

Amazon SQS menyediakan fitur dan kemampuan berikut.

Topik

- [Metadata pesan](#)
- [Sumber daya yang diperlukan untuk memproses pesan Amazon SQS](#)
- [Daftar paginasi](#)
- [Amazon SQS Alokasi Biaya](#)
- [Amazon SQS pemungutan suara pendek dan panjang](#)
- [Antrian surat mati Amazon SQS](#)
- [Batas waktu visibilitas Amazon SQS](#)
- [Antrean penundaan Amazon SQS](#)
- [Antrian sementara Amazon SQS](#)
- [Pengatur waktu pesan Amazon SQS](#)
- [Mengakses EventBridge Pipa Amazon melalui konsol Amazon SQS](#)
- [Mengelola pesan Amazon SQS besar dengan Extended Client Library dan Amazon Simple Storage Service](#)

Metadata pesan

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 AWS lainnya, seperti AWS X-Ray.

Topik

- [Atribut pesan Amazon SQS](#)
- [Atribut pesan Amazon SQS](#)

Atribut pesan Amazon SQS

Amazon SQS memungkinkan Anda menyertakan metadata terstruktur (seperti cap waktu, data geospasial, dan pengidentifikasi) dengan pesan atribut pesan. Setiap pesan dapat memiliki hingga

10 atribut. Atribut pesan bersifat opsional dan terpisah dari isi pesan (namun, atribut pesan bersifat opsional dan terpisah dari isi pesan. Konsumen dapat menggunakan atribut 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 \(konsol\)](#).

Note

Jangan membingungkan atribut pesan dengan Atribut sistem pesan: Sedangkan 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 AWS lainnya, seperti AWS X-Ray.

Topik

- [Komponen atribut pesan](#)
- [Jenis data atribut pesan](#)
- [Menghitung pesan MD5 digest untuk atribut pesan](#)

Komponen atribut pesan

Important

Semua komponen atribut pesan disertakan dalam pembatasan ukuran pesan 256 KB. ParameterName, Type, Value, dan isi 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-z, 0-9, garis bawah (_), tanda hubung (-), dan periode (.). Pembatasan berikut berlaku:
 - Panjang dapat mencapai 256 karakter
 - Tidak dapat dimulai dengan AWS . atau Amazon . (atau variasi casing)
 - Peka huruf besar/kecil
 - Harus unik di antara semua nama atribut untuk pesan
 - Tidak boleh dimulai atau diakhiri dengan periode
 - Tidak harus memiliki periode secara berurutan

- **Jenis**— Tipe data atribut pesan. Tipe yang didukung meliputi `String`, `Number`, dan `Binary`. Anda juga dapat menambahkan informasi khusus untuk semua jenis data. Tipe data memiliki batasan pesan (untuk informasi selengkapnya, lihat [SendMessage](#) di dalam Referensi API Amazon Simple Queue Service). Selain itu, batasan berikut berlaku:
 - Panjang dapat mencapai 256 karakter
 - Peka huruf besar/kecil
- **Nilai**— Nilai atribut pesan. Untuk `String` jenis data, nilai atribut memiliki batasan yang sama dengan isi pesan.

Jenis data atribut pesan

Jenis 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 jenis data logis `String`, `Number`, dan `Binary` dengan label tipe data kustom opsional dengan format *.custom-data-type*

- **Tali**— `String` atribut dapat menyimpan teks Unicode menggunakan karakter XML yang valid.
- **Nomor**— `Number` atribut dapat menyimpan nilai-nilai numerik positif atau negatif. Angka dapat memiliki hingga 38 digit presisi, dan dapat antara 10^{-128} dan 10^{126} .

Note

Amazon SQS menghapus angka nol terdepan dan trailing.

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

Note

Amazon SQS tidak menafsirkan, memvalidasi, atau menggunakan data yang ditambahkan. Label jenis khusus memiliki batasan pesan.

Menghitung pesan MD5 digest untuk atribut pesan

Jika Anda menggunakan AWS SDK for Java, Anda dapat melewati bagian ini.

`ParameterMessageMD5ChecksumHandler` kelas SDK for Java mendukung mencerna pesan MD5 untuk atribut pesan Amazon SQS.

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

Note

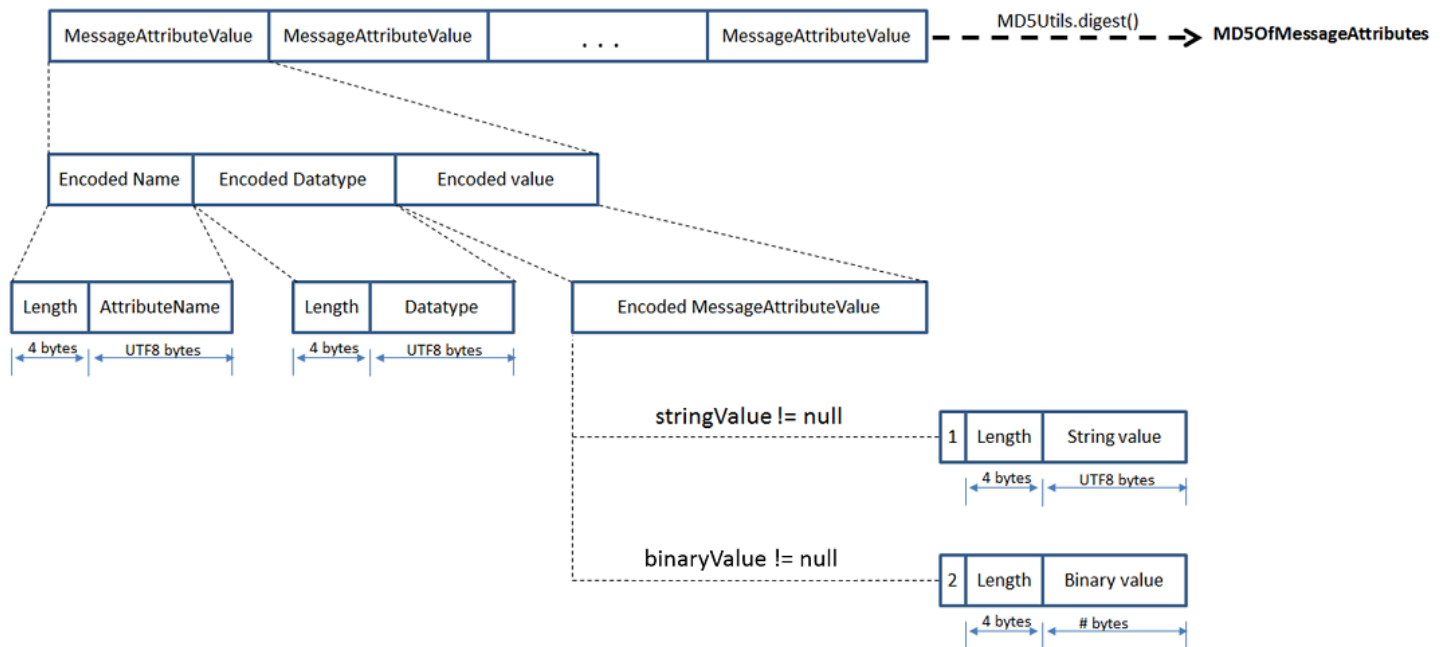
Selalu sertakan sufiks tipe data khusus dalam perhitungan pesan dicerna MD5.

Gambaran Umum

Berikut ini adalah ikhtisar dari algoritma perhitungan pesan MD5 digest:

1. Urutkan semua atribut pesan berdasarkan nama dalam urutan menaik.
2. Mengkodekan masing-masing bagian dari setiap atribut (`Name`, `Type`, dan `Value`) menjadi penyangga.
3. Menghitung pesan mencerna seluruh buffer.

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



Untuk mengkodekan atribut pesan Amazon SQS tunggal

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

Note

Jenis data `logisString` dan `Number` menggunakan `String` jenis transportasi.
Tipe data `logisBinary` menggunakan `Binary` jenis transportasi.

- a. Untuk `String` jenis transportasi, encode 1.
 - b. Untuk `Binary` jenis transportasi, encode 2.
4. Encode nilai atribut.
 - a. Untuk `String` jenis transportasi, mengkodekan nilai atribut: panjang (4 byte) dan UTF-8 byte dari nilai.
 - b. Untuk `Binary` jenis transportasi, mengkodekan nilai atribut: panjang (4 byte) dan byte mentah dari nilai.

Atribut pesan Amazon SQS

Sedangkan 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 lainnya AWS layanan, seperti AWS X-Ray. Untuk informasi selengkapnya, lihat [MessageSystemAttribute](#) permintaan parameter [SendMessage](#) dan [SendMessageBatch](#) Tindakan API, [AWS Trace Header](#) atribut [ReceiveMessage](#) Aksi API, dan [MessageSystemAttributeValue](#) Tipe data di Referensi API Amazon Simple Queue Service.

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

- Saat ini, atribut sistem pesan hanya didukung [AWS Trace Header](#). Jenisnya harus `String` dan nilainya harus diformat dengan benar [AWS X-Ray](#) melacak string sundulan.
- Ukuran atribut sistem pesan tidak dihitung terhadap ukuran total pesan.

Sumber daya yang diperlukan untuk memproses pesan Amazon SQS

Untuk membantu Anda memperkirakan sumber daya yang Anda butuhkan untuk memproses pesan antrian, Amazon SQS dapat menentukan perkiraan jumlah pesan yang tertunda, terlihat, dan tidak terlihat dalam antrian. Untuk informasi selengkapnya tentang visibilitas, lihat [Batas waktu visibilitas Amazon SQS](#).

Note

Untuk antrian standar, hasilnya adalah perkiraan karena arsitektur terdistribusi Amazon SQS. Dalam kebanyakan kasus, hitungan harus mendekati jumlah pesan aktual dalam antrian. Untuk antrian FIFO, hasilnya tepat.

Tabel berikut mencantumkan nama atribut untuk digunakan dengan [GetQueueAttributes](#) tindakan:

Tugas	Nama atribut
Dapatkan perkiraan jumlah pesan yang tersedia untuk diambil dari antrian Anda.	<code>ApproximateNumberOfMessagesVisible</code>
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 <code>delay</code> .	<code>ApproximateNumberOfMessagesDelayed</code>
Dapatkan perkiraan jumlah pesan yang sedang terbang. Pesan dianggap dalam penerbangan jika mereka telah dikirim ke klien tetapi belum dihapus atau belum mencapai akhir jendela visibilitas mereka.	<code>ApproximateNumberOfMessagesNotVisible</code>

Daftar paginasi

Parameter `listQueues` dan `listDeadLetterQueues` Metode 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 dalam setiap respon.

Set parameter `MaxResults` di dalam [listQueues](#) atau [listDeadLetterQueues](#) meminta untuk menentukan jumlah hasil yang akan dikembalikan dalam respons. Jika Anda tidak mengatur `MaxResults`, respon mencakup maksimal 1.000 hasil dan `NextToken` nilai dalam respons adalah null.

Jika Anda menetapkan `MaxResults`, respon mencakup nilai untuk `NextToken` jika ada hasil tambahan untuk ditampilkan. Gunakan `NextToken` sebagai parameter dalam permintaan Anda berikutnya untuk `listQueues` untuk menerima halaman hasil berikutnya. Jika tidak ada hasil tambahan untuk ditampilkan, `NextToken` nilai dalam respons adalah null.

Amazon SQS Alokasi Biaya

Untuk mengelola dan mengidentifikasi antrian Amazon SQS untuk alokasi biaya, Anda dapat menambahkan tanda metadata yang mengidentifikasi tujuan, pemilik, atau lingkungan antrian. Ini sangat berguna jika Anda memiliki banyak antrian. Untuk mengonfigurasi tanda menggunakan konsol Amazon SQS, lihat [the section called “Mengkonfigurasi tag untuk antrian”](#)

Anda juga dapat menggunakan tanda untuk mengatur tagihan AWS Anda untuk merefleksikan struktur biaya Anda sendiri. Untuk melakukannya, daftar untuk membuat Akun AWS tagihan Anda menyertakan kunci dan nilai tanda. Untuk informasi selengkapnya, lihat [Menyiapkan Laporan Alokasi Biaya Bulanan](#) di Panduan Pengguna AWS Billing.

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

Antrean	Kunci	Nilai
MyQueueA	QueueType	Production
MyQueueB	QueueType	Testing

Note

Saat Anda menggunakan tag, ingatlah panduan berikut:

- Kami tidak menyarankan untuk menambahkan lebih dari 50 tag ke antrian. Penandaan mendukung karakter Unicode dalam UTF-8.
- Tag tidak memiliki makna semantik. Amazon SQS menafsirkan tag sebagai string karakter.
- Tanda peka huruf besar dan 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, [kiriman permintaan](#).

Untuk daftar lengkap pembatasan tanda, lihat [Kuota](#).

Amazon SQS pemungutan suara pendek dan panjang

Amazon SQS menyediakan pemungutan suara singkat dan polling panjang untuk menerima pesan dari antrian. Secara default, antrian menggunakan polling singkat.

Dengan pemungutan suara singkat, yang [ReceiveMessage](#) permintaan query hanya subset dari server (berdasarkan distribusi acak tertimbang) untuk menemukan pesan yang tersedia untuk disertakan dalam respon. Amazon SQS mengirimkan respon segera, bahkan jika kueri tidak menemukan pesan.

Dengan polling panjang, yang [ReceiveMessage](#) permintaan query semua server untuk pesan. Amazon SQS mengirimkan respons setelah mengumpulkan setidaknya satu pesan yang tersedia, hingga jumlah maksimum pesan yang ditentukan dalam permintaan. Amazon SQS mengirimkan respons kosong hanya jika waktu tunggu pemungutan suara berakhir.

Bagian berikut menjelaskan rincian pemungutan suara singkat dan polling panjang.

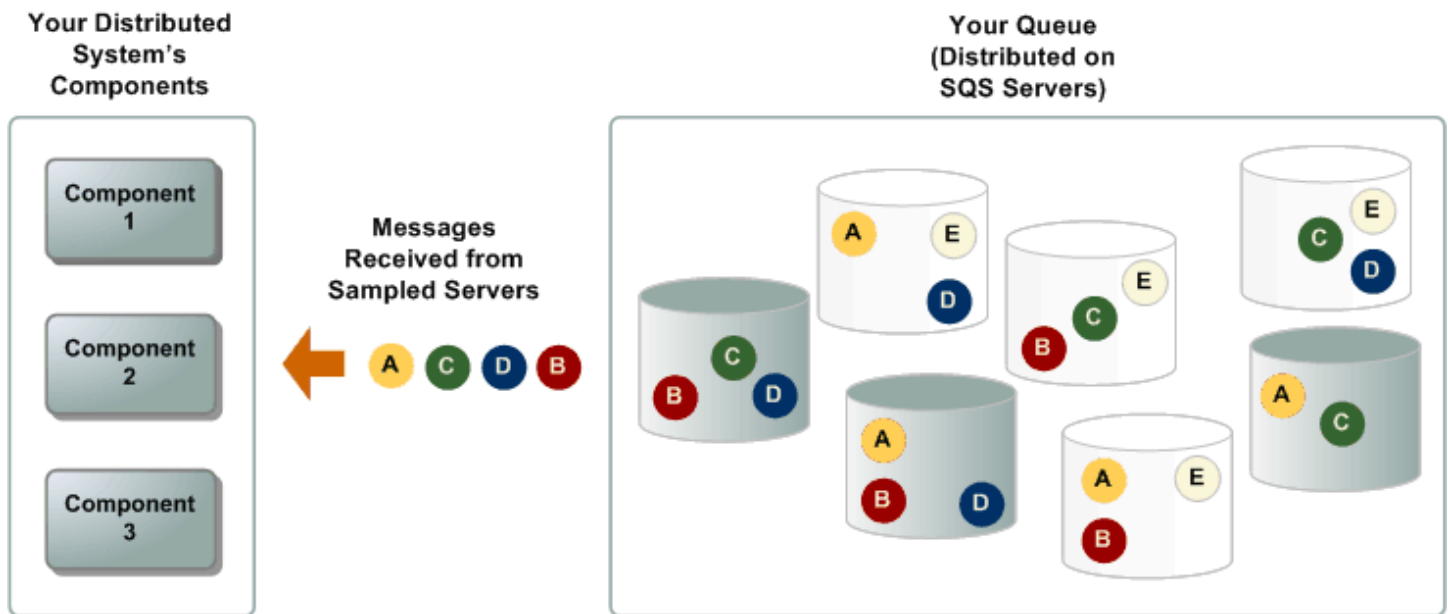
Topik

- [Mengonsumsi pesan menggunakan pemungutan suara singkat](#)
- [Mengonsumsi pesan menggunakan polling panjang](#)
- [Perbedaan antara pemungutan suara panjang dan pendek](#)

Mengonsumsi pesan menggunakan pemungutan suara singkat

Saat Anda mengonsumsi pesan dari antrian menggunakan polling singkat, Amazon SQS mencicipi subset servernya (berdasarkan distribusi acak tertimbang) dan mengembalikan pesan hanya dari server tersebut. Dengan demikian, tertentu [ReceiveMessage](#) permintaan 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 mengonsumsi antrian Anda, Amazon SQS mencicipi semua servernya, dan Anda menerima semua pesan Anda.

Diagram berikut menunjukkan perilaku pendek-polling pesan yang dikembalikan dari antrian standar setelah salah satu komponen sistem Anda membuat permintaan menerima. Amazon SQS mencicipi 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.



Mengonsumsi pesan menggunakan polling panjang

Ketika waktu tunggu untuk `ReceiveMessage` Tindakan API lebih besar dari 0, polling panjang berlaku. Waktu tunggu maksimum pemungutan suara panjang adalah 20 detik. Jajak pendapat panjang membantu mengurangi biaya penggunaan Amazon SQS dengan menghilangkan jumlah respons kosong (bila tidak ada pesan yang tersedia untuk `ReceiveMessageRequest`) dan tanggapan kosong palsu (bila pesan tersedia tetapi tidak disertakan dalam respons). Untuk informasi tentang mengaktifkan polling panjang untuk antrian baru atau yang sudah ada menggunakan konsol Amazon SQS, lihat [Mengkonfigurasi parameter antrian \(konsol\)](#). 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 koneksi kali keluar, respon 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 tanggapan kosong bahkan ketika antrian masih berisi pesan, terutama jika Anda menentukan nilai rendah untuk `ReceiveMessageWaitTimeSeconds` parameter.
- Kurangi tanggapan kosong palsu dengan menanyakan semua—bukan subset server Amazon SQS.
- Kembali pesan segera setelah mereka menjadi tersedia.

Untuk informasi tentang cara mengonfirmasi bahwa antrian kosong, lihat [Mengonfirmasi bahwa antrian kosong](#).

Perbedaan antara pemungutan suara panjang dan pendek

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

- Parameter `ReceiveMessage` set `WaitTimeSeconds` kepada 0.
- Parameter `ReceiveMessage` panggilan tidak diatur `WaitTimeSeconds`, tapi atribut antrian `ReceiveMessageWaitTimeSeconds` diatur ke 0.

Antrian surat mati Amazon SQS

Amazon SQS mendukung antrian huruf mati (DLQ), yang antrian lain (antrian sumber) dapat menargetkan pesan yang tidak dapat diproses (dikonsumsi) dengan sukses. Antrian surat mati berguna untuk men-debug aplikasi atau sistem pesan Anda karena memungkinkan Anda mengisolasi pesan yang tidak dikonsumsi untuk menentukan mengapa pemrosesannya tidak berhasil. Untuk informasi tentang mengonfigurasi antrian huruf mati menggunakan konsol Amazon SQS, lihat [Mengonfigurasi antrian huruf mati \(konsol\)](#). Setelah Anda men-debug aplikasi konsumen atau aplikasi konsumen tersedia untuk mengkonsumsi pesan, Anda dapat menggunakan [kemampuan redrive antrian huruf mati](#) untuk memindahkan pesan kembali ke antrian sumber.

Important

Amazon SQS tidak membuat antrian huruf mati secara otomatis. Anda harus terlebih dahulu membuat antrian sebelum menggunakannya sebagai antrian huruf mati.

Topik

- [Bagaimana cara kerja antrian surat mati?](#)
- [Apa manfaat antrian surat mati?](#)
- [Bagaimana jenis antrian yang berbeda menangani kegagalan pesan?](#)
- [Kapan saya harus menggunakan antrian huruf mati?](#)
- [Memindahkan pesan dari antrian surat mati](#)
- [Memecahkan masalah antrian surat mati](#)

- [Mengkonfigurasi antrian huruf mati \(konsol\)](#)
- [Mengonfigurasi redrive antrian huruf mati](#)
- [CloudTrail persyaratan pembaruan dan izin untuk rean surat mati \(DLQ\) Amazon SQS](#)

Bagaimana cara kerja antrean surat mati?

Terkadang, pesan tidak dapat diproses karena berbagai kemungkinan masalah, seperti kondisi yang salah dalam aplikasi produsen atau konsumen atau perubahan status tak terduga yang menyebabkan masalah dengan kode aplikasi Anda. Misalnya, jika pengguna menempatkan pesanan web dengan ID produk tertentu, tetapi ID produk dihapus, kode toko web gagal dan menampilkan kesalahan, dan pesan dengan permintaan pesanan dikirim ke antrian huruf mati.

Kadang-kadang, produsen dan konsumen mungkin gagal menafsirkan aspek protokol yang mereka gunakan untuk berkomunikasi, menyebabkan korupsi atau kehilangan pesan. Selain itu, kesalahan perangkat keras konsumen dapat merusak muatan pesan.

Kebijakan redrive menentukan antrian sumber, antrian huruf mati, dan kondisi di mana Amazon SQS memindahkan pesan dari yang pertama ke yang terakhir jika konsumen antrian sumber gagal memproses pesan beberapa kali tertentu. `maxReceiveCount` ini adalah berapa kali konsumen mencoba menerima pesan dari antrian tanpa menghapusnya sebelum dipindahkan ke antrian huruf mati. Menyetel `maxReceiveCount` ke nilai rendah seperti 1 akan mengakibatkan kegagalan untuk menerima pesan yang menyebabkan pesan dipindahkan ke antrian huruf mati. Kegagalan tersebut termasuk kesalahan jaringan dan kesalahan ketergantungan klien. 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. Kebijakan ini berlaku untuk antrian surat mati potensial. Anda dapat memilih apakah akan mengizinkan semua antrian sumber, mengizinkan antrian sumber tertentu, atau menolak semua antrian sumber. Standarnya adalah mengizinkan semua antrian sumber 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.

Untuk menentukan antrian huruf mati, Anda dapat menggunakan konsol atau SDK. AWS Anda harus melakukan ini untuk setiap antrian yang mengirim pesan ke antrian huruf mati. Beberapa antrian dari jenis yang sama dapat menargetkan antrian huruf mati tunggal. Untuk informasi lebih lanjut, lihat

Mengkonfigurasi antrian huruf mati (konsol) `RedrivePolicy` dan `RedriveAllowPolicy` atribut dari `CreateQueue` atau `SetQueueAttributes` tindakan.

Important

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

Anda harus menggunakan yang sama Akun AWS untuk membuat antrian huruf mati dan antrian lain yang mengirim pesan ke antrian surat mati. Juga, antrian surat mati harus berada di wilayah yang sama dengan antrian lain yang menggunakan antrian huruf mati. Misalnya, jika Anda membuat antrian di wilayah AS Timur (Ohio) dan Anda ingin menggunakan antrian huruf mati dengan antrian itu, antrian kedua juga harus berada di wilayah AS Timur (Ohio). 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.

Apa manfaat antrian surat mati?

Tugas utama antrian huruf mati adalah menangani siklus hidup pesan yang tidak dikonsumsi. Antrian surat mati memungkinkan Anda menyisihkan dan mengisolasi pesan yang tidak dapat diproses dengan benar untuk menentukan mengapa pemrosesannya tidak berhasil. Menyiapkan antrian surat mati memungkinkan Anda melakukan hal berikut:

- Konfigurasi alarm untuk pesan apa pun yang dipindahkan ke antrian huruf mati.
- Periksa log untuk pengecualian yang mungkin menyebabkan pesan dipindahkan ke antrian huruf mati.

- Menganalisis isi pesan yang dipindahkan ke antrian surat mati untuk mendiagnosis perangkat lunak atau masalah perangkat keras produsen atau konsumen.
- Tentukan apakah Anda telah memberi konsumen Anda waktu yang cukup untuk memproses pesan.

Bagaimana jenis antrian yang berbeda menangani kegagalan pesan?

Antrian standar

[Antrian standar](#) terus memproses pesan hingga berakhirnya periode retensi. Pemrosesan pesan yang berkelanjutan ini meminimalkan kemungkinan antrian Anda diblokir oleh pesan yang tidak dapat diproses. Pemrosesan pesan berkelanjutan juga memberikan pemulihan yang lebih cepat untuk antrian Anda.

Dalam sistem yang memproses ribuan pesan, memiliki sejumlah besar pesan yang berulang kali gagal diakui dan dihapus oleh konsumen dapat meningkatkan biaya dan menempatkan beban ekstra pada perangkat keras. Daripada mencoba memproses pesan yang gagal sampai kedaluwarsa, lebih baik memindahkannya ke antrian surat mati setelah beberapa upaya pemrosesan.

Note

Antrian standar memungkinkan sejumlah besar pesan dalam penerbangan. Jika sebagian besar pesan Anda tidak dapat dikonsumsi dan tidak dikirim ke antrian surat mati, tingkat pemrosesan pesan yang valid dapat melambat. Jadi, untuk menjaga efisiensi antrian Anda, pastikan aplikasi Anda menangani pemrosesan pesan dengan benar.

Antrian FIFO

[Antrian FIFO](#) menyediakan pemrosesan tepat sekali dengan mengonsumsi pesan secara berurutan dari grup pesan. Dengan demikian, 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.

 Note


Antrian FIFO memungkinkan jumlah pesan dalam penerbangan yang lebih rendah. Jadi, agar antrian FIFO Anda tidak diblokir oleh pesan, pastikan aplikasi Anda menangani pemrosesan pesan dengan benar.

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.

Kapan saya harus menggunakan antrian huruf mati?



Gunakan antrian huruf mati dengan antrian standar. Anda harus selalu memanfaatkan antrian surat mati ketika aplikasi Anda tidak bergantung pada pemesanan. Antrian surat mati dapat membantu Anda memecahkan masalah operasi transmisi pesan yang salah.

 Note

Bahkan ketika Anda menggunakan antrian huruf mati, Anda harus terus memantau antrian Anda dan mencoba lagi mengirim pesan yang gagal karena alasan sementara.



Gunakan antrian huruf mati untuk mengurangi jumlah pesan dan untuk mengurangi kemungkinan mengekspos sistem Anda ke pesan pil racun (pesan yang dapat diterima tetapi tidak dapat diproses).



Jangan gunakan antrian huruf mati dengan antrian standar ketika Anda ingin dapat terus mencoba kembali transmisi pesan tanpa batas waktu. Misalnya, jangan gunakan antrian huruf mati jika program Anda harus menunggu proses dependen menjadi aktif atau tersedia.



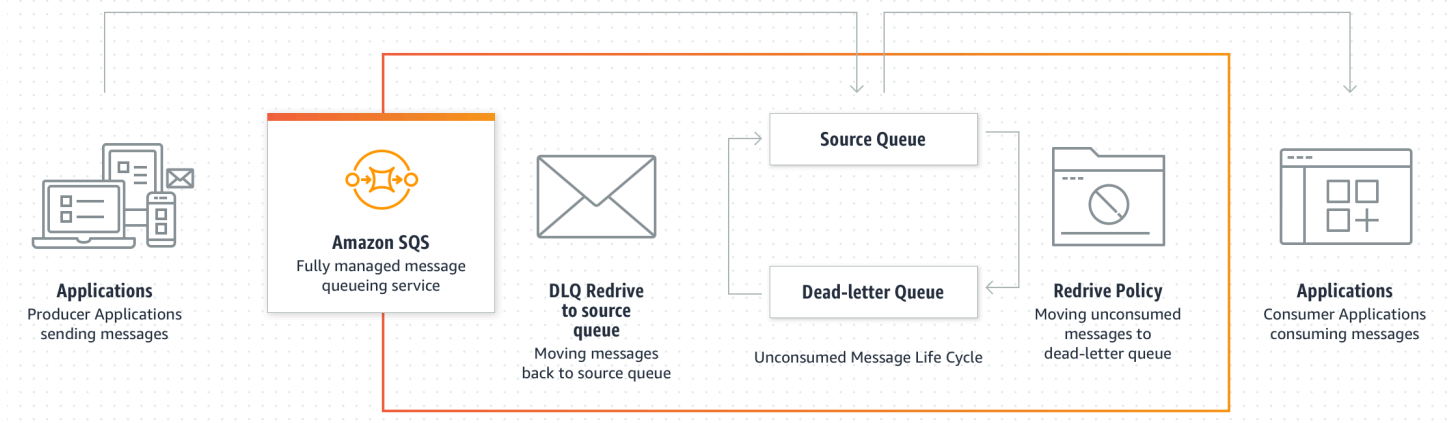
Jangan gunakan antrian huruf mati dengan antrian FIFO jika Anda tidak ingin merusak urutan pesan atau operasi yang tepat. Misalnya, jangan gunakan antrian huruf mati dengan instruksi dalam Daftar

Keputusan Edit (EDL) untuk rangkaian pengeditan video, di mana mengubah urutan pengeditan mengubah konteks pengeditan berikutnya.

Memindahkan pesan dari antrian surat mati

Anda dapat menggunakan penggerak ulang antrian huruf mati untuk mengelola siklus hidup pesan yang tidak dikonsumsi. Setelah Anda menyelidiki atribut dan metadata terkait yang tersedia untuk pesan yang tidak dikonsumsi dalam antrian huruf mati standar atau FIFO, Anda dapat mengarahkan ulang pesan kembali ke antrian sumbernya. Penggerak ulang antrian huruf mati mengurangi penagihan panggilan API dengan mengelompokkan pesan saat memindahkannya.

Tugas redrive menggunakan Amazon SQS `SendMessageBatch`, `ReceiveMessage`, dan `DeleteMessageBatch` API atas nama pengguna untuk menggerakkan ulang pesan. Oleh karena itu, semua pesan yang digerakkan ulang dianggap sebagai pesan baru `messageId` dengan periode `enqueueTime` retensi dan baru. [Harga redrive antrian huruf mati menggunakan jumlah panggilan API yang dipanggil dan tagihan berdasarkan harga Amazon SQS.](#)



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. Untuk petunjuk tentang mengonfigurasi redrive antrian huruf mati, lihat. [Mengonfigurasi redrive antrian huruf mati](#)

Note

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.

Ketika pesan digerakkan ulang dari antrian huruf mati FIFO, pesan `GroupId` dan `DeDuplicationId` tetap sama, dan pesan menerima `MessageId` baru.

Antrian surat mati Amazon SQS 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 terhubung dengan pesan baru dari produser.

Memecahkan masalah antrian surat mati

Dalam beberapa kasus, antrian surat mati Amazon SQS mungkin tidak selalu berperilaku seperti yang diharapkan. Bagian ini memberikan gambaran umum dari masalah umum dan menunjukkan cara mengatasinya.

Melihat pesan menggunakan konsol dapat menyebabkan pesan dipindahkan ke antrian huruf mati

Amazon SQS menghitung melihat pesan di konsol terhadap kebijakan redrive antrian terkait. Jadi, jika Anda melihat pesan di konsol berapa kali ditentukan dalam kebijakan redrive antrian yang sesuai, pesan akan dipindahkan ke antrian huruf mati antrian yang sesuai.

Untuk menyesuaikan perilaku ini, Anda dapat melakukan salah satu hal berikut:

- Meningkatkan pengaturan Penerima Maksimum untuk kebijakan redrive antrian 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. Dengan demikian, adalah mungkin untuk nilai-nilai `NumberOfMessagesSent` dan `NumberOfMessagesReceived` untuk menjadi berbeda.

Untuk informasi tentang membuat dan mengonfigurasi redrive antrian huruf mati

Perhatikan bahwa redrive antrian huruf mati mengharuskan Anda menetapkan izin yang sesuai untuk Amazon SQS untuk menerima pesan dari antrian huruf mati dan mengirim pesan ke antrian tujuan. Jika izin tidak mencukupi, antrian huruf mati redrive ke antrian sumber tidak memulai redrive pesan dan dapat gagal tugas. Anda dapat melihat status tugas redrive pesan Anda untuk memperbaiki masalah dan mencoba lagi.

Topik

- [Mengkonfigurasi antrian huruf mati \(konsol\)](#)
- [Mengonfigurasi redrive antrian huruf mati](#)
- [CloudTrail persyaratan pembaruan dan izin untuk rean surat mati \(DLQ\) Amazon SQS](#)

Mengkonfigurasi antrian huruf mati (konsol)

Antrian huruf mati adalah antrian yang dapat digunakan oleh satu atau lebih antrian sumber untuk pesan yang tidak berhasil dikonsumsi. Untuk informasi selengkapnya, lihat [Antrian surat mati 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 \(konsol\)](#)

Antrian dead-letter pada antrian FIFO juga harus merupakan antrian FIFO. Demikian pula, antrian dead-letter dari antrian standar juga harus menjadi antrian 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/](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.

Mengonfigurasi redrive antrian huruf mati

Anda dapat mengonfigurasi redrive antrian huruf mati untuk memindahkan pesan standar yang tidak digunakan dari antrian huruf mati yang ada kembali ke antrian sumbernya. Untuk informasi selengkapnya tentang pengaktifan ulang antrian surat mati, lihat [Memindahkan pesan dari antrian surat mati](#)

Mengonfigurasi redrive antrian huruf mati untuk antrian standar (API) yang ada


Anda dapat mengonfigurasi redrive antrian huruf mati menggunakan tindakan API berikut.

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 antrian standar yang ada (konsol)

1. [Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/](https://console.aws.amazon.com/sqs/).
2. Di panel navigasi, pilih Antrian.
3. Pilih nama antrian yang telah Anda konfigurasi 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.

 Note

Antrian tujuan kustom harus sesuai dengan jenis antrian huruf mati. Misalnya, jika antrian huruf mati adalah antrian FIFO, maka antrian tujuan khusus harus berupa antrian FIFO juga.

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.

Tugas redrive mengatur ulang periode retensi. Yang baru messageID dan enqueueTime ditugaskan ke pesan yang digerakkan ulang.

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 antrian tujuan.

Mengkonfigurasi izin antrian untuk redrive antrian huruf mati

Anda dapat memberi pengguna akses ke tindakan antrian 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	<ul style="list-style-type: none"> Tambah <code>sqs:StartMessageMoveTask</code> , <code>sqs:ReceiveMessage</code> , <code>sqs:DeleteMessage</code> , dan <code>sqs:GetQueueAttributes</code> antrian huruf mati. Jika antrian huruf mati atau antrian sumber asli dienkripsi (juga dikenal sebagai antrian SSE), <code>kms:Decrypt</code> untuk kunci KMS apa pun yang telah digunakan untuk mengenkripsi pesan juga diperlukan. Tambahkan antrian tujuan. <code>sqs:SendMessage</code> Jika antrian tujuan dienkripsi, <code>kms:GenerateDataKey</code> dan <code>kms:Decrypt</code> juga diperlukan.
Untuk membatalkan redrive pesan yang sedang berlangsung	<ul style="list-style-type: none"> Tambah <code>sqs:CancelMessageMoveTask</code> , <code>sqs:ReceiveMessage</code> , <code>sqs:DeleteMessage</code> , dan <code>sqs:GetQueueAttributes</code> antrian huruf mati. Jika antrian huruf mati dienkripsi (juga dikenal sebagai antrian SSE), juga diperlukan. <code>kms:Decrypt</code>
Untuk menampilkan status pemindahan pesan	<ul style="list-style-type: none"> Tambahkan <code>sqs:ListMessageMoveTasks</code> dan <code>sqs:GetQueueAttributes</code> 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`
 - ResourceARN 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"
    ],
    "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:ReceiveMessage`

- sqs:DeleteMessage
- sqs:GetQueueAttributes
- Resource<DLQ_region><DLQ_accountId><DLQ_name>ARN dari antrian huruf mati (misalnya, "arn:aws:sqs::: ").
- sqs:SendMessage
- ResourceARN 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"
      ],
      "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 rean surat mati (DLQ) Amazon SQS

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

- [CloudTrail penggantian nama acara untuk rean surat mati](#)
- [Izin yang diperbarui untuk rean surat mati antrian surat mati](#)

CloudTrail penggantian nama acara

Pada tanggal 15 Oktober 2023, CloudTrail nama acara untuk rean surat mati akan berubah di konsol Amazon SQS. Jika Anda telah mengatur alarm untuk ini CloudTrail acara, Anda harus memperbaruinya sekarang. Berikut ini adalah yang baru CloudTrail nama acara 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
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 konfigurasi menggunakan AWSKonsol hanya di wilayah di mana Anda sebelumnya telah menggunakan redrive DLQ. Misalnya, Anda memiliki "Akun A" di us-east-1 dan eu-west-1. "Akun A" digunakan untuk menggerakkan ulang pesan di AWSKonsol 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 AWSKonsol 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 AWSKonsol. Namun, jika Anda menggunakan fitur redrive DLQ pada AWSKonsol 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 [antrean terenkripsi \(SSE\) sisi server](#), yang sesuai AWS KMS izin kunci diperlukan.

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 terkirim

Kebijakan terkelola berikut berisi izin diperbarui yang diperlukan:

- **AmazonSQSFullAccess**— Termasuk tugas redrive antrian huruf mati berikut: mulai, batalkan, dan daftar.
- **AmazonSQSReadOnlyAccess**— Menyediakan akses hanya-baca, dan termasuk daftar tugas redrive antrian huruf mati.

Step 1

Add permissions

Step 2

Review

Add permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

Permissions options

Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

Copy permissions
Copy all group memberships, attached managed policies, inline policies, and any existing permissions boundaries from an existing user.

Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Permissions policies (1/1051)

2 matches

	Policy name	Type	Attached entities
<input checked="" type="checkbox"/>	AmazonSQSFullAccess	AWS managed	0
<input type="checkbox"/>	AmazonSQSReadOnly...	AWS managed	0

Cancel Next

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

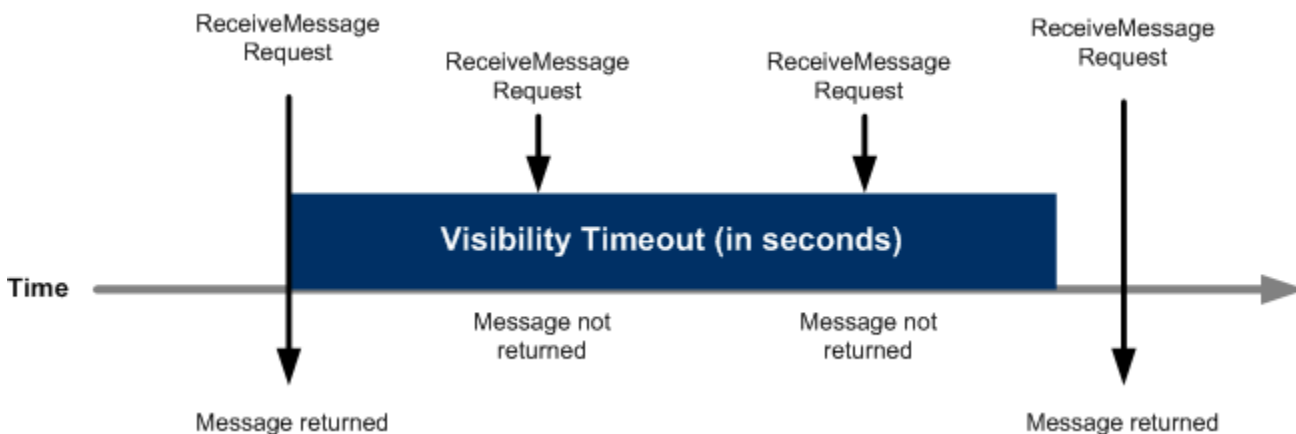
Note

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

1. Masuk ke AWS CloudTrail konsol.
2. Pada Riwayat acara halaman, di bawah Cari atribut, gunakan menu tarik-turun untuk memilih Nama acara. Kemudian, cari `CreateMoveTask`.
3. Pilih acara untuk membuka Detail halaman. Di Catatan acara bagian, ambil `UserName` atau `RoleName` dari `userIdentityARN`.
4. Masuk ke konsol IAM.
 - Untuk pengguna, pilih Pengguna. Pilih pengguna dengan `UserName` terkirim pada langkah sebelumnya.
 - Untuk peran, pilih Peran. Cari pengguna dengan `RoleName` terkirim pada langkah sebelumnya.
5. Pada Detail halaman, di izin bagian, tinjau kebijakan apa pun dengan `sqs:awalan diAction`, atau meninjau kebijakan yang memiliki antrean Amazon SQS yang ditentukan `Resource`.

Batas waktu visibilitas Amazon SQS

Ketika konsumen menerima dan memproses pesan dari antrian, pesan tetap dalam antrian. Amazon SQS tidak secara otomatis menghapus pesan. 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 tunggu visibilitas untuk antrian menggunakan konsol, lihat [Mengkonfigurasi parameter antrian \(konsol\)](#).

Note

Untuk antrian standar, batas waktu visibilitas bukanlah jaminan terhadap 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 produser 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, konsumen dapat mencoba lagi sebanyak yang diperlukan, menggunakan ID upaya permintaan terima yang sama. Dengan asumsi bahwa konsumen menerima setidaknya satu pengakuan sebelum batas waktu visibilitas berakhir, beberapa percobaan ulang tidak memengaruhi pemesanan pesan.
- Bila Anda menerima pesan dengan ID grup pesan, tidak ada lagi pesan untuk ID grup pesan yang sama yang ditampilkan kecuali jika Anda menghapus pesan atau pesan tersebut akan terlihat.

Topik

- [Dalam pesan penerbangan](#)
- [Mengatur batas waktu visibilitas](#)
- [Mengubah batas waktu visibilitas untuk pesan](#)
- [Mengakhiri batas waktu visibilitas untuk pesan](#)

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 antrean.

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

Important

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

Untuk sebagian besar antrian standar (tergantung pada lalu lintas antrian dan backlog pesan), dapat 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 pesan `OverLimit` kesalahan. Jika Anda menggunakan [polling panjang](#), Amazon SQS tidak mengembalikan pesan kesalahan. Untuk menghindari kuota, Anda harus menghapus pesan dari antrean setelah diproses. Anda juga dapat meningkatkan jumlah antrian yang Anda gunakan untuk memproses pesan Anda. Untuk meminta kenaikan kuota, [kiriman permintaan dukungan](#).

Untuk antrian FIFO, bisa ada maksimal 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 antrean 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 tambahan, lihat [Cara Kerja API FIFO Amazon SQS](#).

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 selama 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 menetapkan batas waktu visibilitas khusus untuk pesan yang dikembalikan tanpa mengubah batas waktu antrian keseluruhan. Untuk informasi selengkapnya lihat praktik terbaik di [Memproses pesan pada waktu yang tepat](#) bagian.

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 kemudian—selama konsumen Anda masih mengerjakan pesan—terus perpanjang batas waktu visibilitas hingga 2 menit setiap menit.

Important

Batas waktu visibilitas maksimum adalah 12 jam sejak Amazon SQS menerima `ReceiveMessage` permintaan tersebut. 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 dengan `VisibilityTimeout` sama dengan 43.200 detik, kemungkinan akan gagal. Namun, menggunakan nilai 43.195 detik akan bekerja kecuali ada penundaan yang signifikan antara meminta pesan

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

Mengubah batas waktu visibilitas untuk pesan

Ketika Anda menerima pesan dari antrean dan mulai memprosesnya, batas waktu visibilitas untuk antrean 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 [ChangeMessageVisibility](#) tindakan.

Misalnya, jika batas waktu default untuk antrean adalah 60 detik, 15 detik telah berlalu sejak Anda menerima pesan, dan Anda mengirim `ChangeMessageVisibility` panggilan dengan `VisibilityTimeout` set ke 10 detik, 10 detik mulai dihitung dari waktu Anda melakukan `ChangeMessageVisibility` panggilan. Dengan demikian, setiap upaya untuk mengubah batas waktu visibilitas atau menghapus pesan itu 10 detik setelah Anda 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 memengaruhi batas waktu penerimaan pesan atau antrean nanti.

Mengakhiri batas waktu visibilitas untuk pesan

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

Untuk menghentikan batas waktu visibilitas pesan setelah menelepon `ReceiveMessage`, panggil [ChangeMessageVisibility](#) dengan `VisibilityTimeout` set ke 0 detik.

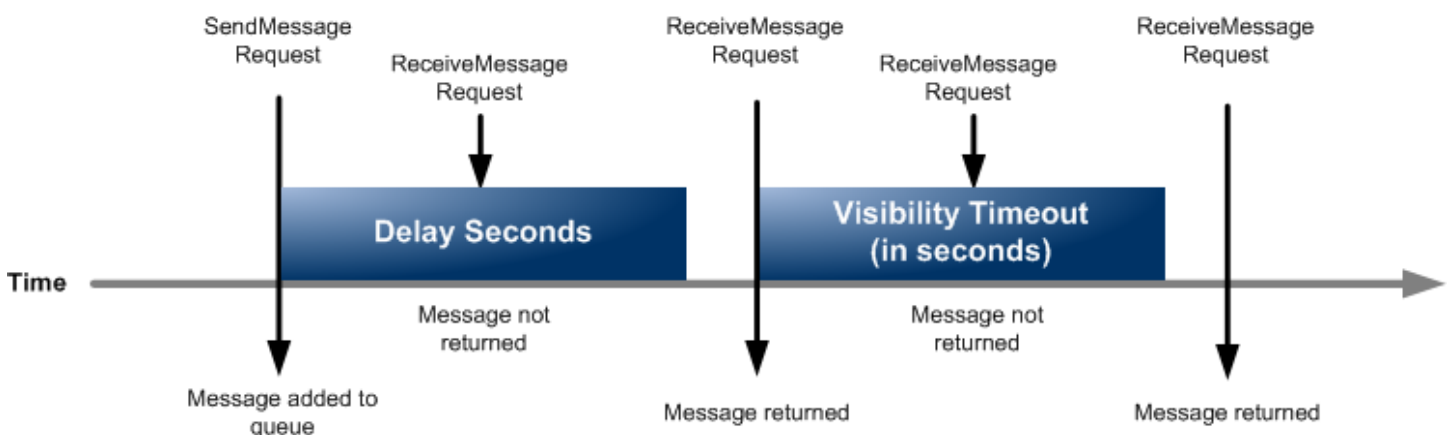
Antrean penundaan Amazon SQS

Antrian penundaan 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 antrian penundaan, setiap pesan 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 mengkonfigurasi antrian penundaan menggunakan konsol, lihat [Mengkonfigurasi parameter antrian \(konsol\)](#).

Note

Untuk antrian standar, pengaturan penundaan per antrian adalah tidak surut—changing pengaturan tidak memengaruhi penundaan pesan yang sudah ada dalam antrian. Untuk antrian FIFO, pengaturan penundaan per antrian adalah retroaktif—changing pengaturan memengaruhi penundaan pesan yang sudah ada dalam antrian.

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



Untuk mengatur detik penundaan pesan individu, bukan pada seluruh antrian, gunakan [timer pesan](#) untuk memungkinkan Amazon SQS menggunakan pengatur waktu pesan `DelaySeconds` nilai alih-alih antrian `delayDelaySeconds` nilai.

Antrian sementara Amazon SQS

Antrian sementara membantu Anda menghemat waktu pengembangan dan biaya penerapan saat menggunakan pola pesan umum seperti permintaan-tanggapan. Anda dapat menggunakan [Klien Antrian Sementara](#) untuk membuat antrian sementara dengan throughput tinggi, hemat biaya, dan dikelola aplikasi.

Klien memetakan beberapa antrian sementara—antrian terkelola 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](#)
- [Pola pesan permintaan-respons \(antrian virtual\)](#)
- [Contoh skenario: Memproses permintaan login](#)
 - [Di sisi klien](#)
 - [Di sisi server](#)
- [Membersihkan antrian](#)

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 yang terkait dengan pesan](#).

TheAmazonSQSVirtualQueuesClient kelas wrapper menambahkan dukungan untuk atribut yang terkait dengan antrian virtual. Untuk membuat antrian virtual, Anda harus memanggil `CreateQueue` tindakan 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
```

Ketika seorang produser memanggil `SendMessage` atau `SendMessageBatch` tindakan 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.

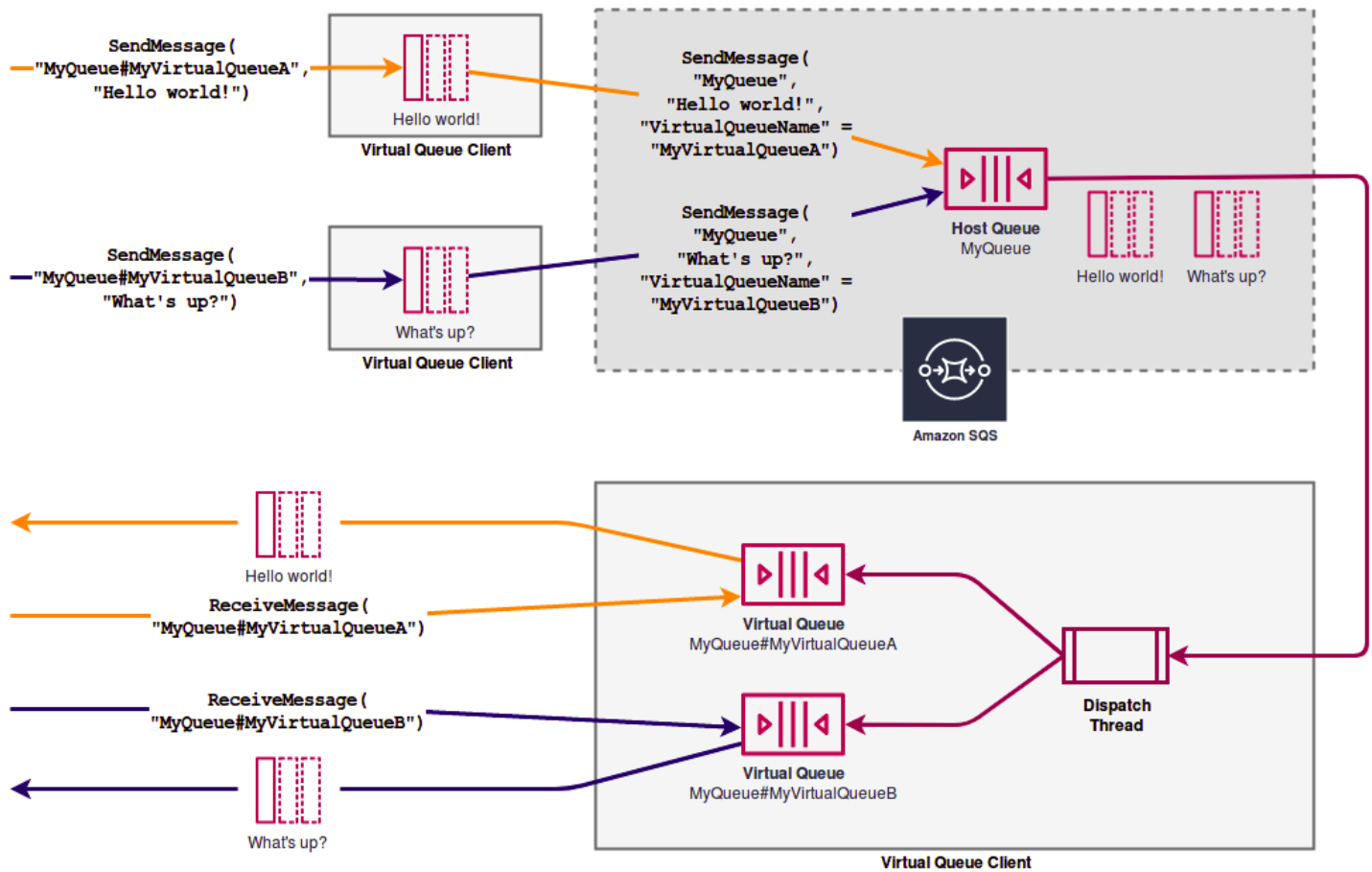
Sementara konsumen menyebut `ReceiveMessage` tindakan API pada URL antrian virtual, Klien Antrian Sementara memblokir panggilan secara lokal hingga utas latar belakang mengirim pesan ke antrian virtual. (Proses ini mirip dengan pengambilan pesan di [Klien Asinkron Buffered](#): satu tindakan API dapat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

The `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 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 termasuk `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 permintaan-tanggapan pola pesan, 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 respons permintaan](#).

Diagram berikut memperlihatkan konfigurasi umum menggunakan pola ini.



Contoh skenario: Memproses permintaan login

Contoh berikut memperlihatkan 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()` adalah 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 membutuhkan Klien Antrian Sementara, itu harus memanggil `shutdown()` metode. 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 `TagQueueTindakan` API untuk menandai antrian yang masih digunakan.

Note

Setiap tindakan API yang diambil pada antrian menandainya sebagai non-idle, termasuk `receiveMessage` tindakan yang tidak mengembalikan pesan.

Thread latar belakang menggunakan `ListQueues` dan `ListTags` Tindakan API 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 antrian. Penundaan default (minimum) untuk pesan adalah 0 detik. Maksimal 15 menit. Untuk informasi tentang mengirim pesan dengan timer menggunakan konsol, lihat [Kirim 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 antrian 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 antrian Amazon SQS ditentukan sebagai sumber pipa. Untuk informasi selengkapnya, lihat [Membuat EventBridge pipa](#) di Panduan EventBridge Pengguna Amazon.

Important

Pesan pada antrian Amazon SQS dibaca oleh satu pipa dan kemudian dihapus dari antrian setelah diproses, apakah pesan tersebut cocok dengan filter yang dapat Anda konfigurasi 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 antrian 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) 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 antrian Amazon SQS Amazon.

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
- Mengambil objek pesan dari bucket S3
- Hapus objek pesan dari bucket S3

Prasyarat

Contoh berikut menggunakan AWS Java SDK. Untuk menginstal dan mengatur SDK, lihat [Menyiapkan AWS SDK for Java](#) di AWS SDK for Java Panduan Developer.

Sebelum Anda menjalankan kode contoh, konfigurasi AWS kredensial Anda. Untuk informasi selengkapnya, lihat [Menyiapkan Kredensial dan Wilayah AWS untuk Pengembangan](#) di Panduan Developer AWS SDK for Java.

[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

AWSSDK for Java 1.x Contoh: Menggunakan Amazon S3 untuk mengelola pesan Amazon SQS besar

Contoh AWSSDK 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 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
```



```
*
* https://aws.amazon.com/apache2.0
*
* or in the "license" file accompanying this file. This file is distributed
* on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
* express or implied. See the License for the specific language governing
* permissions and limitations under the License.
*
*/

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

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

public class SQSExtendedClientExample {

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

    public static void main(String[] args) {

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

        /*
        * Set the Amazon S3 bucket name, and then set a lifecycle rule on the
        * bucket to permanently delete objects 14 days after each object's
        * creation date.
        */
    }
}
```

```
*/
final BucketLifecycleConfiguration.Rule expirationRule =
    new BucketLifecycleConfiguration.Rule();
expirationRule.withExpirationInDays(14).withStatus("Enabled");
final BucketLifecycleConfiguration lifecycleConfig =
    new BucketLifecycleConfiguration().withRules(expirationRule);

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

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

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

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

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

// Send the message.
final SendMessageRequest myMessageRequest =
    new SendMessageRequest(myQueueUrl, myLongString);
```

```
sqExtended.sendMessage(myMessageRequest);
System.out.println("Sent the message.");

// Receive the message.
final ReceiveMessageRequest receiveMessageRequest =
    new ReceiveMessageRequest(myQueueUrl);
List<Message> messages = sqExtended
    .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();
sqExtended.deleteMessage(new DeleteMessageRequest(myQueueUrl,
    messageReceiptHandle));
System.out.println("Deleted the message.");

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

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

private static void deleteBucketAndAllContents(AmazonS3 client) {

    ObjectListing objectListing = client.listObjects(S3_BUCKET_NAME);

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

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

```

```
        break;
    }
}

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

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

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

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

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

import com.amazon.sqs.javamessaging.AmazonSQSExtendedClient;
import com.amazon.sqs.javamessaging.ExtendedClientConfiguration;
```

```
import org.joda.time.DateTime;
import org.joda.time.format.DateTimeFormat;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.BucketLifecycleConfiguration;
import software.amazon.awssdk.services.s3.model.CreateBucketRequest;
import software.amazon.awssdk.services.s3.model.DeleteBucketRequest;
import software.amazon.awssdk.services.s3.model.DeleteObjectRequest;
import software.amazon.awssdk.services.s3.model.ExpirationStatus;
import software.amazon.awssdk.services.s3.model.LifecycleExpiration;
import software.amazon.awssdk.services.s3.model.LifecycleRule;
import software.amazon.awssdk.services.s3.model.LifecycleRuleFilter;
import software.amazon.awssdk.services.s3.model.ListObjectVersionsRequest;
import software.amazon.awssdk.services.s3.model.ListObjectVersionsResponse;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Response;
import software.amazon.awssdk.services.s3.model.PutBucketLifecycleConfigurationRequest;
import software.amazon.awssdk.services.sqs.SqsClient;
import software.amazon.awssdk.services.sqs.model.CreateQueueRequest;
import software.amazon.awssdk.services.sqs.model.CreateQueueResponse;
import software.amazon.awssdk.services.sqs.model.DeleteMessageRequest;
import software.amazon.awssdk.services.sqs.model.DeleteQueueRequest;
import software.amazon.awssdk.services.sqs.model.Message;
import software.amazon.awssdk.services.sqs.model.ReceiveMessageRequest;
import software.amazon.awssdk.services.sqs.model.ReceiveMessageResponse;
import software.amazon.awssdk.services.sqs.model.SendMessageRequest;

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

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

    public static void main(String[] args) {

        /*
         * Create a new instance of the builder with all defaults (credentials
```

```
    * and region) set automatically. For more information, see
    * Creating Service Clients in the AWS SDK for Java Developer Guide.
    */
final S3Client s3 = S3Client.create();

/*
 * Set the Amazon S3 bucket name, and then set a lifecycle rule on the
 * bucket to permanently delete objects 14 days after each object's
 * creation date.
 */
final LifecycleRule lifeCycleRule = LifecycleRule.builder()
    .expiration(LifecycleExpiration.builder().days(14).build())
    .filter(LifecycleRuleFilter.builder().prefix("").build())
    .status(ExpirationStatus.ENABLED)
    .build();
final BucketLifecycleConfiguration lifecycleConfig =
BucketLifecycleConfiguration.builder()
    .rules(lifeCycleRule)
    .build();

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

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

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

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

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

// Create a message queue for this example
```

```
    final String queueName = "MyQueue-" + UUID.randomUUID();
    final CreateQueueResponse createQueueResponse =
sqsExtended.createQueue(CreateQueueRequest.builder().queueName(queueName).build());
    final String myQueueUrl = createQueueResponse.queueUrl();
    System.out.println("Queue created.");

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

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

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

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

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

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

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

}

private static void deleteBucketAndAllContents(S3Client client) {
```

```

    ListObjectsV2Response listObjectsResponse =
client.listObjectsV2(ListObjectsV2Request.builder().bucket(S3_BUCKET_NAME).build());

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

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

    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()).build());
    });

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 [AWSHalaman](#) beranda, lalu pilih Buat AWS Akun. Ikuti petunjuk online. [Untuk informasi tentang kredensial, lihat Kredensial.](#)

- AWSSDK: 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. Default-nya adalah `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
```

Mengkonfigurasi antrian Amazon SQS (konsol)

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 \(ABAC\) untuk Amazon SQS](#)
- [Mengkonfigurasi parameter antrian \(konsol\)](#)
- [kebijakan akses \(konsol\)](#)
- [Mengkonfigurasi enkripsi sisi server \(SSE\) untuk antrian menggunakan kunci enkripsi yang dikelola SQS \(konsol\)](#)
- [Mengkonfigurasi enkripsi sisi server \(SSE\) untuk antrian \(konsol\)](#)
- [Mengkonfigurasi tag alokasi biaya untuk antrian Amazon SQS \(konsol\)](#)
- [Berlangganan antrean Amazon SQS ke topik Amazon SNS](#)
- [Mengkonfigurasi antrian untuk memicu AWS Lambda fungsi \(konsol\)](#)
- [Mengirim pesan dengan atribut \(konsol\)](#)

Kontrol akses berbasis atribut (ABAC) 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 mengotorisasi 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 antrian Amazon SQS. Untuk mempelajari selengkapnya tentang penandaan AWS, lihat [Strategi AWS Penandaan](#) dan [Amazon SQS Alokasi Biaya](#)

Note

ABAC untuk Amazon SQS saat ini tersedia di AWS semua Wilayah Komersil tempat Amazon SQS tersedia, dengan pengecualian berikut:

- Asia Pasifik (Hyderabad)
- Asia Pasifik (Melbourne)
- Eropa (Spanyol)
- Eropa (Zurich)

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 menskalakan 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](#)
- [Penandaan untuk kontrol akses](#)
- [Membuat pengguna IAM dan antrian Amazon SQS](#)
- [Menguji kontrol akses berbasis atribut](#)

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 antrian 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 dengan kunci tag dalam kebijakan	Kebijakan antrian dan kebijakan IAM	TagQueue , UntagQueue , CreateQueue

Penandaan untuk kontrol akses

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: Tanda.
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 antrian 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 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 antrian 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 (konsol)

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 pemungutan suara pendek dan panjang](#).
- Aktifkan deduplikasi berbasis konten — Amazon SQS dapat secara otomatis membuat ID deduplikasi berdasarkan isi pesan. Untuk informasi selengkapnya, lihat [Memulai antrian Amazon SQS FIFO](#).
- 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. Lihat informasi yang lebih lengkap di [Throughput tinggi untuk antrian FIFO](#) dan [Kuota yang terkait dengan pesan](#).
- Kebijakan redrive allow: mendefinisikan antrian sumber mana yang dapat menggunakan antrian ini sebagai antrian huruf mati. Untuk informasi selengkapnya, lihat [Antrian surat mati Amazon SQS](#).

Untuk mengonfigurasi parameter antrian untuk antrian yang ada (konsol)

1. [Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/](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. Lihat informasi yang lebih lengkap di [Throughput tinggi untuk antrian FIFO](#) dan [Kuota yang terkait dengan pesan](#).

11. Untuk kebijakan Izinkan Remove, 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.

kebijakan akses (konsol)

Saat [mengedit](#) antrian, Anda dapat mengonfigurasi kebijakan aksesnya.

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 mengkonfigurasi kebijakan akses 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 Kebijakan akses.
5. Edit pernyataan kebijakan akses di kotak masukan. Untuk informasi selengkapnya tentang pernyataan kebijakan akses, lihat [Manajemen identitas dan akses di Amazon SQS](#).
6. Setelah selesai mengonfigurasi kebijakan akses, pilih Simpan.

Mengkonfigurasi enkripsi sisi server (SSE) untuk antrian menggunakan kunci enkripsi yang dikelola SQS (konsol)

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/](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.

Mengkonfigurasi enkripsi sisi server (SSE) untuk antrian (konsol)

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, lihat [Enkripsi diam](#).

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/](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 AWSKunci 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 Amazon SQS (konsol)

Untuk membantu mengatur dan mengidentifikasi antrian Amazon SQS, Anda dapat menambahkan tag alokasi biaya ke dalamnya. Untuk informasi selengkapnya, lihat [Amazon SQS Alokasi Biaya](#).

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/](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 antrean Amazon SQS ke topik Amazon SNS

Anda dapat berlangganan antrean Amazon SQS ke topik Amazon Simple Notification Service (Amazon SNS). Ketika Anda mempublikasikan pesan ke topik, Amazon SNS mengirimkan pesan ke masing-masing antrean berlangganan. Amazon SQS mengelola langganan dan izin yang diperlukan. Untuk informasi selengkapnya tentang Amazon SNS, lihat [Apa yang Dimaksud dengan Amazon SNS?](#) di Panduan Developer Amazon Simple Notification Service.

Ketika Anda berlangganan antrean Amazon SQS ke topik SNS, Amazon SNS menggunakan HTTPS untuk meneruskan pesan ke Amazon SQS. Untuk informasi tentang penggunaan Amazon SNS dengan antrean Amazon SQS terenkripsi, lihat [Konfigurasi izin KMS untuk layanan AWS](#).

⚠ Important

Amazon SQS mendukung maksimum 20 pernyataan per kebijakan akses. Berlangganan topik Amazon SNS menambahkan satu pernyataan tersebut. 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 topik Tentukan 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 dan kemudian masukkan Amazon Resource Name (ARN) topik.

6. Pilih Save (Simpan).
7. Untuk memverifikasi hasil langganan, publikasikan topik dan kemudian lihat pesan yang dikirim topik ke antrian. Untuk informasi selengkapnya, lihat [penerbitan pesan Amazon SNS](#) di Panduan Developer Amazon Simple Notification Service.

Jika antrian Amazon SQS dan topik SNS Anda berbeda Akun AWS, pemilik topik harus mengonfirmasi langganan terlebih dahulu. 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 antrian atau AWS Lambda fungsi Amazon SQS di Wilayah yang berbeda](#) dalam Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon

Mengkonfigurasi antrian untuk memicu AWS Lambda fungsi (konsol)

Anda dapat menggunakan AWS Lambda fungsi untuk memproses pesan dalam antrian Amazon SQS. Lambda melakukan polling pada antrian 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 konfigurasi 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/](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.

Mengirim pesan dengan atribut (konsol)

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

Untuk mengirim pesan dengan atribut ke 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.
4. Pilih Kirim dan menerima pesan.
5. Masukkan parameter atribut pesan.

- a. Dalam kotak teks nama, masukkan nama unik hingga 256 karakter.
- b. Untuk jenis atribut, pilih **Tali**, **Nomor**, atau **Biner**.
- c. (Opsional) Masukkan tipe data khusus. Misalnya, Anda dapat menambahkan **byte**, **int**, atau **float** sebagai tipe data kustom untuk **Nomor**.
- d. Dalam kotak teks nilai, masukkan nilai atribut pesan.

The screenshot shows a dialog box titled "Message attributes - Optional" with an "Info" icon. It contains a form with four input fields: "Enter name", a dropdown menu set to "String", "Custom type", and "Enter value". Below these fields is a button labeled "Add new attribute".

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

The screenshot shows the same dialog box as above, but with an additional "Remove" button to the right of the second attribute row. The "Add new attribute" button is still present at the bottom.

7. Anda dapat mengubah nilai atribut kapan saja sebelum mengirim pesan.
8. Untuk menghapus atribut, pilih **Menghapus**. 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 **Melihat detail**. Memiilih **Selesai** tutup **Detail pesan** kotak dialog.

Praktik terbaik untuk Amazon SQS

Praktik terbaik dapat membantu Anda memaksimalkan Amazon SQS.

Topik

- [Rekomendasi untuk standar Amazon SQS dan antrean FIFO](#)
- [Rekomendasi tambahan untuk antrean FIFO Amazon SQS](#)

Rekomendasi untuk standar Amazon SQS dan antrean FIFO

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

Topik

- [Bekerja dengan pesan Amazon SQS](#)
- [Mengurangi biaya Amazon SQS](#)
- [Berpindah dari antrean Amazon SQS Standard ke antrean FIFO](#)

Bekerja dengan pesan Amazon SQS

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

Topik

- [Memproses pesan pada waktu yang tepat](#)
- [Menangani kesalahan permintaan](#)
- [Menyiapkan polling panjang](#)
- [Menangkap pesan bermasalah](#)
- [Menyiapkan retensi antrian huruf mati](#)
- [Menghindari pemrosesan pesan yang tidak konsisten](#)
- [Menerapkan sistem respons permintaan](#)

Memproses pesan pada waktu yang tepat

Mengatur batas waktu visibilitas tergantung pada berapa lama aplikasi Anda memproses dan menghapus pesan. Misalnya, jika aplikasi Anda membutuhkan waktu 10 detik untuk memproses pesan dan Anda menetapkan 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 membutuhkan 10 detik untuk memproses pesan tetapi Anda menetapkan batas waktu visibilitas hanya 2 detik, pesan duplikat diterima oleh konsumen lain sementara 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 kemudian—selama konsumen Anda masih mengerjakan pesan—terus perpanjang batas waktu visibilitas hingga 2 menit setiap menit.

Important

Batas waktu visibilitas maksimum adalah 12 jam sejak Amazon SQS menerima permintaan tersebut `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 dengan `VisibilityTimeout` sama dengan 43.200 detik, kemungkinan akan gagal. Namun, menggunakan nilai 43.195 detik akan bekerja kecuali ada penundaan yang signifikan antara meminta pesan melalui `ReceiveMessage` dan memperbarui batas waktu visibilitas. Jika konsumen Anda membutuhkan 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 lebih lanjut, lihat [Error Retries dan Exponential Backoff](#) di dalam. AWS Referensi Umum Amazon Web Services
- Jika Anda tidak menggunakan fitur AWS SDK untuk coba ulang dan backoff, izinkan jeda (misalnya, 200 ms) sebelum mencoba ulang [ReceiveMessage](#) tindakan setelah tidak menerima pesan, batas waktu, atau pesan kesalahan dari Amazon SQS. Untuk penggunaan selanjutnya [ReceiveMessage](#) yang memberikan hasil yang sama, memungkinkan jeda lagi (misalnya, 400 ms).

Menyiapkan polling panjang

Ketika waktu tunggu untuk tindakan [ReceiveMessage](#) API lebih besar dari 0, polling panjang berlaku. Waktu tunggu polling panjang maksimum 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 (bila pesan tersedia tetapi tidak disertakan dalam respons). Untuk informasi selengkapnya, lihat [Amazon SQS pemungutan suara pendek dan panjang](#).

Untuk pemrosesan pesan yang optimal, gunakan strategi berikut:

- Dalam kebanyakan kasus, Anda dapat mengatur waktu [ReceiveMessage](#) tunggu hingga 20 detik. Jika 20 detik terlalu lama untuk aplikasi Anda, tetapkan waktu [ReceiveMessage](#) tunggu yang lebih singkat (minimum 1 detik). Jika Anda tidak menggunakan AWS SDK untuk mengakses Amazon SQS, atau jika Anda mengonfigurasi AWS SDK agar memiliki waktu tunggu 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 thread untuk setiap antrian, bukan satu thread untuk semua antrian. Menggunakan thread tunggal untuk setiap antrian memungkinkan aplikasi Anda untuk memproses pesan di setiap antrian saat tersedia, sementara menggunakan thread tunggal untuk polling beberapa antrian dapat menyebabkan aplikasi Anda menjadi tidak dapat memproses pesan yang tersedia dalam antrian lain sementara aplikasi menunggu (hingga 20 detik) untuk antrian yang tidak memiliki pesan yang tersedia.

⚠ Important

Untuk menghindari kesalahan HTTP, pastikan bahwa 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, konfigurasi antrian huruf [mati](#).

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

Menyiapkan retensi antrian huruf 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, anggap bahwa pesan menghabiskan 1 hari dalam antrian asli sebelum dipindahkan ke antrian huruf mati. Jika periode retensi antrian huruf mati adalah 4 hari, pesan akan dihapus dari antrian huruf mati setelah 3 hari dan adalah 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 adalah 4 hari. `ApproximateAgeOfOldestMessage`

Menghindari pemrosesan pesan yang tidak konsisten

Karena Amazon SQS adalah sistem terdistribusi, dimungkinkan bagi konsumen untuk tidak menerima pesan bahkan ketika Amazon SQS menandai pesan sebagai terkirim saat kembali berhasil dari panggilan metode `ReceiveMessage` API. Dalam hal ini, Amazon SQS mencatat pesan yang dikirimkan setidaknya sekali, meskipun konsumen belum pernah menerimanya. Karena tidak ada upaya tambahan untuk mengirimkan pesan yang dilakukan dalam kondisi ini, kami tidak menyarankan untuk mengatur jumlah penerimaan maksimum ke 1 untuk antrean [huruf mati](#).

Menerapkan sistem respons permintaan

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

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

Untuk informasi selengkapnya tentang menerapkan pola respons permintaan menggunakan Klien Antrian Sementara, 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, batch 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 batching permintaan, gunakan polling panjang bersama dengan klien asinkron [buffer yang disertakan](#) dengan AWS SDK for Java

Note

Amazon SQS Buffered Asynchronous Client saat ini tidak mendukung antrean FIFO.

Menggunakan modus polling yang sesuai

- Polling panjang memungkinkan Anda menggunakan pesan dari antrean Amazon SQS segera setelah tersedia.
 - Untuk mengurangi biaya penggunaan Amazon SQS dan mengurangi jumlah penerimaan kosong ke antrean kosong (respons terhadap `ReceiveMessage` tindakan yang tidak menampilkan pesan), aktifkan polling panjang. Untuk informasi selengkapnya, lihat [Polling Panjang Amazon SQS](#).
 - Untuk meningkatkan efisiensi saat polling untuk beberapa thread dengan beberapa penerima, mengurangi jumlah thread.
 - Pemungutan suara panjang lebih disukai daripada pemungutan suara pendek dalam banyak kasus.
- Polling singkat segera mengembalikan respons, meskipun antrian Amazon SQS yang disurvei kosong.
 - Untuk memenuhi persyaratan aplikasi yang mengharapkan tanggapan langsung terhadap `ReceiveMessage` permintaan tersebut, gunakan polling singkat.
 - Pemungutan suara pendek ditagih sama dengan polling panjang.

Berpindah dari antrean Amazon SQS Standard ke antrean FIFO

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

Untuk informasi selengkapnya, lihat [Pindah dari antrian standar ke antrian FIFO](#).

Rekomendasi tambahan untuk antrean FIFO Amazon SQS

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 [Amazon Simple Queue Service API Reference](#).

Topik

- [Menggunakan ID deduplikasi pesan Amazon SQS](#)
- [Menggunakan ID grup pesan Amazon SQS](#)
- [Menggunakan ID upaya permintaan terima Amazon SQS](#)

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, setiap pesan yang dikirim dengan ID deduplikasi pesan yang sama berhasil diterima tetapi tidak terkirim selama interval deduplikasi 5 menit.

Note

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

Menyediakan 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 dianggap unik oleh Amazon SQS.
- 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, coba lagi jumlah yang disertakan dalam isi pesan) yang harus diperlakukan Amazon SQS sebagai duplikat.

Mengaktifkan deduplikasi untuk sistem produsen/konsumen tunggal

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

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

- Meskipun konsumen tidak diharuskan memberikan ID upaya permintaan terima untuk setiap permintaan, ini adalah praktik terbaik karena memungkinkan urutan percobaan gagal untuk mengeksekusi lebih cepat.
- Anda dapat mencoba lagi mengirim atau menerima permintaan karena tidak mengganggu pemesanan 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.

- Produser 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 pesannya 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 `ChangeMessageVisibility` tindakan. Namun, jika batas waktu visibilitas berakhir, konsumen lain dapat segera mulai memproses pesan, menyebabkan pesan diproses beberapa kali. Untuk menghindari skenario ini, konfigurasi [antrian huruf mati](#).

Bekerja dengan batas waktu visibilitas

Untuk performa 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 termasuk dalam 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 dalam urutan yang ketat).

Interleaving beberapa grup pesan yang diurutkan

Untuk interleave beberapa grup pesan yang diurutkan dalam antrian FIFO tunggal, 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 multiple-produsen/konsumen

Untuk menghindari pemrosesan pesan duplikat dalam sistem dengan beberapa produsen dan konsumen di mana throughput dan latensi lebih penting daripada memesan, produsen harus membuat 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 pengiriman 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 maksimal 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. Antrian FIFO melihat melalui pesan 20 ribu 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 [antrean huruf mati](#).

Hindari menggunakan kembali ID grup pesan yang sama dengan antrian virtual

Untuk mencegah pesan dengan ID grup pesan yang sama yang dikirim ke antrean [virtual yang berbeda dengan antrian](#) host yang sama saling memblokir, hindari menggunakan kembali ID grup pesan yang sama dengan antrian virtual.

Menggunakan ID upaya permintaan terima Amazon SQS

ID upaya 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 upaya permintaan terima dan coba lagi dengan ID upaya 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 antrian 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 \(SSE\)](#)
- [Mengkonfigurasi tag untuk antrian](#)
- [Atribut pesan](#)

Menggunakan enkripsi sisi server (SSE)

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

Menambahkan SSE ke antrian yang sudah ada

Untuk mengaktifkan enkripsi sisi server untuk antrian yang sudah ada, gunakan enkripsi sisi server [SetQueueAttributes](#) metode untuk mengatur `KmsMasterKeyId` atribut.

Contoh kode berikut menetapkan AWS KMS key sebagai AWS kunci KMS terkelola untuk Amazon SQS. Contoh ini juga menetapkan [AWS KMS key periode penggunaan kembali](#) 140 detik.

Sebelum menjalankan kode contoh, pastikan bahwa Anda telah mengatur AWS kredensial. Untuk informasi selengkapnya, lihat [Menyiapkan Kredensial dan Wilayah AWS untuk Pengembangan](#) di Panduan Developer AWS SDK for Java 2.x.

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

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

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

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

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

sqsClient.setQueueAttributes(set_attrs_request);
```

Menonaktifkan SSE untuk antrian

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

Important

`null` bukanlah nilai yang valid untuk `KmsMasterKeyId`.

Membuat antrian dengan SSE

Untuk mengaktifkan SSE saat membuat antrian, tambahkan `KmsMasterKeyId` atribut ke [CreateQueue](#) Metode API.

Contoh berikut membuat antrian baru dengan SSE diaktifkan. Antrian menggunakan AWS kunci KMS terkelola untuk Amazon SQS. Contoh ini juga menetapkan [AWS KMS key periode penggunaan kembali](#) 160 detik.

Sebelum menjalankan kode contoh, pastikan bahwa Anda telah mengatur AWS kredensial. Untuk informasi selengkapnya, lihat [Menyiapkan Kredensial dan Wilayah AWS untuk Pengembangan](#) di Panduan Developer AWS SDK for Java 2.x.

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

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

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

Mengambil atribut SSE

Untuk informasi tentang mengambil atribut antrian, lihat [Contoh](#) di dalam Referensi Amazon Simple Queue Service.

Untuk mengambil ID kunci KMS atau periode penggunaan kembali kunci data untuk antrian tertentu, jalankan [GetQueueAttributes](#) metode dan mengambil `KmsMasterKeyId` dan `KmsDataKeyReusePeriodSeconds` nilai-nilai.

Mengkonfigurasi tag untuk antrian

Gunakan tag alokasi biaya untuk membantu mengatur dan mengidentifikasi antrian Amazon SQS Anda. Contoh berikut menunjukkan cara mengonfigurasi tag dengan menggunakan AWS SDK for Java. Untuk informasi selengkapnya, lihat [Amazon SQS Alokasi Biaya](#).

Sebelum menjalankan kode contoh, pastikan bahwa Anda telah mengatur AWS kredensi. Untuk informasi selengkapnya, lihat [Menyiapkan Kredensial dan Wilayah AWS untuk Pengembangan](#) di Panduan Developer AWS SDK for Java 2.x.

Mencantumkan tanda

Untuk membuat daftar tag untuk antrian, gunakan `ListQueueTags` Metode.

```
// 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 menambah atau memperbarui nilai tag untuk antrian, gunakan `TagQueue` Metode.

```
// 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 membuang satu tanda atau lebih dari antrian, gunakan `tandaUntagQueueMetode`. Contoh berikut menghapus `Accounting ID` Tanda.

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

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

Atribut pesan

Anda dapat menyertakan metadata terstruktur (seperti cap waktu, data geospasial, tanda tangan, dan pengidentifikasi) dengan pesan atribut pesan. Untuk informasi selengkapnya, lihat [Atribut pesan Amazon SQS](#).

Sebelum Anda menjalankan kode contoh, pastikan bahwa Anda telah mengatur AWS kredensial. Untuk informasi selengkapnya, lihat [Menyiapkan Kredensial dan Wilayah AWS untuk Pengembangan](#) di Panduan Developer AWS SDK for Java 2.x.

Mendefinisikan atribut

Untuk menentukan atribut untuk pesan, tambahkan kode berikut, yang menggunakan `MessageAttributeValue` tipe data. Untuk informasi selengkapnya, lihat [Komponen atribut pesan](#) dan [Jenis data atribut pesan](#).

Parameter AWS SDK for Java secara otomatis menghitung checksum atribut body dan message message dan membandingkannya dengan data yang dikembalikan oleh Amazon SQS. Untuk informasi selengkapnya, lihat [AWS SDK for Java 2.x Panduan Pengembangan](#) dan [Menghitung pesan MD5 digest untuk atribut pesan](#) untuk bahasa pemrograman lainnya.

String

Contoh ini mendefinisikan `String` atribut `Named` dengan nilainya `Jane`.

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

Number

Contoh ini mendefinisikan `Number` atribut `AccurateWeight` dengan nilainya `230.000000000000000001`.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
messageAttributes.put("AccurateWeight", new MessageAttributeValue()
    .withDataType("Number")
    .withStringValue("230.000000000000000001"));
```

Binary

Contoh ini mendefinisikan `Binary` atribut `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` menamakan `EmployeeId` dengan nilainya `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` menamakan `AccountId` dengan nilainya `000123456`.

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

Note

Karena tipe data dasar `Number`, yang [ReceiveMessage](#) Metode kembali `123456`.

Binary (custom)

Contoh ini mendefinisikan atribut kustom `Binary.JPEG` menamakan `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 bahwa `sendMessage` metode mengeksekusi setelah Anda memberikan ID grup pesan. Jika Anda menggunakan [SendMessageBatch](#) Metode bukan [SendMessage](#), Anda harus menentukan atribut pesan untuk setiap pesan dalam batch.

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.

Perpustakaan mendukung pengiriman dan menerima pesan ke antrian (model point-to-point JMS) sesuai dengan [Spesifikasi JMS 1.1](#). Perpustakaan mendukung pengiriman pesan teks, byte, atau objek secara serentak ke antrian Amazon SQS. Perpustakaan juga mendukung menerima objek secara serentak atau asinkron.

Untuk informasi tentang fitur Perpustakaan Pesan Java Amazon SQS yang mendukung spesifikasi JMS 1.1, lihat [Didukung implementasi JMS 1.1](#) dan [FAQ Amazon SQS](#).

Topik

- [Prasyarat](#)
- [Memulai dengan Perpustakaan Pesan Java Amazon SQS](#)
- [Menggunakan Klien Amazon SQS Java Message Service \(JMS\) dengan klien Amazon SQS lainnya](#)
- [Contoh Java yang berfungsi untuk menggunakan JMS dengan antrian Standar Amazon SQS](#)
- [Didukung implementasi JMS 1.1](#)

Prasyarat

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 Perpustakaan Pesan Java Amazon SQS](#)

Memulai dengan Perpustakaan Pesan Java Amazon SQS

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 termasuk dalam Amazon SQS Java Messaging Library memeriksa apakah antrian Amazon SQS ada. Jika antrian tidak ada, klien akan membuatnya.

Membuat koneksi JMS

1. Buat pabrik koneksi dan panggil `createConnection` terhadap 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();
```

Parameter `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 tidak termasuk dalam spesifikasi JMS, seperti membuat antrian baru. Namun, `getWrappedAmazonSQSClient` metode juga menyediakan versi dibungkus dari klien Amazon SQS yang digunakan oleh koneksi saat ini. Pembungkus mengubah setiap pengecualian dari klien menjadi `JMSEException`, memungkinkan untuk lebih mudah digunakan oleh kode yang ada yang mengharapkan `JMSEException` kejadian.

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

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

- Jika Anda menggunakan `getWrappedAmazonSQSClient`, objek klien kembali 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 akan membuatnya. Jika antrian memang ada, fungsi tidak mengembalikan apa-apa. Untuk informasi selengkapnya, lihat bagian “Buat antrian jika diperlukan” di bagian [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 `.fifo` sufiks.

Untuk informasi lebih lanjut tentang `ContentBasedDeduplication` atribut, lihat [Persis-sekali diproses](#).

Mengirim pesan secara sinkron

1. Ketika koneksi dan antrian Amazon SQS yang mendasarinya siap, buat sesi JMS yang tidak bertransaksi dengan `AUTO_ACKNOWLEDGE` mode.

```
// 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 menetapkan ID grup pesan. Anda juga dapat menetapkan ID pesan. Untuk informasi selengkapnya, lihat [Istilah kunci](#).

```
// 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 serentak

1. Untuk menerima pesan, buat konsumen untuk antrian yang sama dan panggil `start` metode.

Anda dapat menelepon `start` metode pada koneksi kapan saja. Namun, konsumen tidak mulai menerima pesan sampai Anda menyebutnya.

```
// 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 mencetak isi dari pesan yang diterima.

- Setelah menerima pesan dari antrian standar, Anda dapat mengakses isi 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 isi pesan dan atribut pesan khusus FIFO lainnya, seperti ID grup pesan, ID deduplikasi pesan, dan nomor urut. Untuk informasi selengkapnya, lihat [Istilah kunci](#).

```
// 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();
```

Outputnya 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 ini. Untuk informasi tambahan, lihat `SpringExampleConfiguration.xml`, `SpringExample.java`, dan kelas-kelas pembantu lainnya di `ExampleConfiguration.java` dan `ExampleCommon.java` di dalam [Contoh Java yang berfungsi untuk menggunakan JMS dengan antrian Standar Amazon SQS](#) bagian.

Untuk contoh lengkap mengirim dan menerima objek, lihat [TextMessageSender.java](#) dan [SyncMessageReceiver.java](#).

Menerima pesan secara asinkron

Dalam contoh di [Memulai dengan Perpustakaan Pesan Java Amazon SQS](#), pesan dikirim ke `MyQueue` dan diterima serentak.

Contoh berikut menunjukkan cara menerima pesan secara asinkron melalui pendengar.

1. Melaksanakan `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 (JMSEException e) {
            e.printStackTrace();
        }
    }
}
```

Parameter `onMessage` metode `MessageListener` antarmuka dipanggil ketika Anda menerima pesan. Dalam implementasi pendengar ini, teks yang tersimpan dalam pesan dicetak.

2. Alih-alih secara eksplisit memanggil `receive` metode pada konsumen, mengatur pendengar pesan konsumen untuk instance dari `MyListener` implementasi. Benang 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-langkahnya identik dengan yang ada di [Memulai dengan Perpustakaan Pesan Java Amazon SQS](#) contoh. Untuk contoh lengkap konsumen asinkron, lihat `AsyncMessageReceiver.java` di [Contoh Java yang berfungsi untuk menggunakan JMS dengan antrian Standar Amazon SQS](#).

Output untuk contoh ini terlihat serupa dengan yang berikut ini:

```
JMS Message ID:8example-588b-44e5-bbcf-d816example2
Received: Hello World!
```

Menggunakan mode pengakuan klien

Contohnya di [Memulai dengan Perpustakaan Pesan Java Amazon SQS](#) menggunakan `AUTO_ACKNOWLEDGE` mode di mana setiap pesan yang diterima diakui secara otomatis (dan karena itu dihapus dari antrian Amazon SQS yang mendasarinya).

1. Untuk secara eksplisit mengakui pesan setelah diproses, Anda harus membuat sesi dengan `CLIENT_ACKNOWLEDGE` mode.

```
// 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 juga diakui secara implisit. 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-langkahnya identik dengan yang ada di [Memulai dengan Perpustakaan Pesan Java Amazon SQS](#) 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 serupa dengan yang berikut ini:

```
JMS Message ID:4example-aa0e-403f-b6df-5e02example5
Received: Hello World!
Acknowledged: ID:4example-aa0e-403f-b6df-5e02example5
```

Menggunakan mode pengakuan unordered

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 secara individual dan eksplisit diakui oleh klien, terlepas dari pesanan penerimaan mereka. Untuk melakukan ini, buat sesi dengan `UNORDERED_ACKNOWLEDGEMode`.

```
// Create the non-transacted session with UNORDERED_ACKNOWLEDGE mode.
Session session = connection.createSession(false, SQSSession.UNORDERED_ACKNOWLEDGE);
```

Langkah yang tersisa identik dengan yang ada di [Menggunakan mode pengakuan klien](#) contoh. Untuk contoh lengkap konsumen sinkron dengan `UNORDERED_ACKNOWLEDGEMode`, lihat `SyncMessageReceiverUnorderedAcknowledge.java`.

Dalam contoh ini, output terlihat serupa dengan yang berikut ini:

```
JMS Message ID:dexample-73ad-4adb-bc6c-4357example7
Received: Hello World!
Acknowledged: ID:dexample-73ad-4adb-bc6c-4357example7
```

Menggunakan Klien Amazon SQS Java Message Service (JMS) dengan klien Amazon SQS lainnya

Menggunakan Klien Amazon SQS Java Message Service (JMS) dengan AWSSDK membatasi ukuran pesan Amazon SQS menjadi 256 KB. Namun, Anda dapat membuat penyedia JMS menggunakan klien Amazon SQS. Misalnya, Anda dapat menggunakan Klien JMS dengan Amazon SQS Extended Client Library for Java untuk mengirim pesan Amazon SQS yang berisi referensi ke muatan 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 ini 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 ini 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, lihat [Untuk membuat antrian FIFO](#), dan [Mengirim pesan secara sinkron](#) [Menerima pesan secara serentak](#) (Menerima pesan secara sinkron sama untuk antrian standar dan FIFO. Namun, pesan dalam antrian FIFO berisi lebih banyak atribut.)

ExampleConfiguration.java

Contoh kode Java SDK v 1.x berikut menetapkan nama antrian default, wilayah, dan kredensial yang akan digunakan dengan contoh Java lainnya.

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

```
* express or implied. See the License for the specific language governing
* permissions and limitations under the License.
*
*/

public class ExampleConfiguration {
    public static final String DEFAULT_QUEUE_NAME = "SQSJMSClientExampleQueue";

    public static final Region DEFAULT_REGION = Region.getRegion(Regions.US_EAST_2);

    private static String getParameter( String args[], int i ) {
        if( i + 1 >= args.length ) {
            throw new IllegalArgumentException( "Missing parameter for " + args[i] );
        }
        return args[i+1];
    }

    /**
     * Parse the command line and return the resulting config. If the config parsing
     fails
     * print the error and the usage message and then call System.exit
     *
     * @param app the app to use when printing the usage string
     * @param args the command line arguments
     * @return the parsed config
     */
    public static ExampleConfiguration parseConfig(String app, String args[]) {
        try {
            return new ExampleConfiguration(args);
        } catch (IllegalArgumentException e) {
            System.err.println( "ERROR: " + e.getMessage() );
            System.err.println();
            System.err.println( "Usage: " + app + " [--queue <queue>] [--region
<region>] [--credentials <credentials>] ");
            System.err.println( "  or" );
            System.err.println( "      " + app + " <spring.xml>" );
            System.exit(-1);
            return null;
        }
    }

    private ExampleConfiguration(String args[]) {
        for( int i = 0; i < args.length; ++i ) {
            String arg = args[i];

```

```
        if( arg.equals( "--queue" ) ) {
            setQueueName(getParameter(args, i));
            i++;
        } else if( arg.equals( "--region" ) ) {
            String regionName = getParameter(args, i);
            try {
                setRegion(Region.getRegion(Regions.fromName(regionName)));
            } catch( IllegalArgumentException e ) {
                throw new IllegalArgumentException( "Unrecognized region " +
regionName );
            }
            i++;
        } else if( arg.equals( "--credentials" ) ) {
            String credsFile = getParameter(args, i);
            try {
                setCredentialsProvider( new
PropertiesFileCredentialsProvider(credsFile) );
            } catch (AmazonClientException e) {
                throw new IllegalArgumentException("Error reading credentials from
" + credsFile, e );
            }
            i++;
        } else {
            throw new IllegalArgumentException("Unrecognized option " + arg);
        }
    }
}

private String queueName = DEFAULT_QUEUE_NAME;
private Region region = DEFAULT_REGION;
private AWSCredentialsProvider credentialsProvider = new
DefaultAWSCredentialsProviderChain();

public String getQueueName() {
    return queueName;
}

public void setQueueName(String queueName) {
    this.queueName = queueName;
}

public Region getRegion() {
    return region;
}
}
```

```
public void setRegion(Region region) {
    this.region = region;
}

public AWSCredentialsProvider getCredentialsProvider() {
    return credentialsProvider;
}

public void setCredentialsProvider(AWSCredentialsProvider credentialsProvider) {
    // Make sure they're usable first
    credentialsProvider.getCredentials();
    this.credentialsProvider = credentialsProvider;
}
}
```

TextMessageSender.java

Contoh kode Java berikut menciptakan produser pesan teks.

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

public class TextMessageSender {
    public static void main(String args[]) throws JMSEException {
        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 (JMSEException e) {
        System.err.println( "Failed sending message: " + e.getMessage() );
        e.printStackTrace();
    }
}
}
```

SyncMessageReceiver.java

Contoh kode Java berikut menciptakan konsumen pesan sinkron.

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

public class SyncMessageReceiver {
    public static void main(String args[]) throws JMSEException {
        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 (JMSEException e) {
        System.err.println( "Error receiving from SQS: " + e.getMessage() );
        e.printStackTrace();
    }
}
}
```

AsyncMessageReceiver.java

Contoh kode Java berikut menciptakan konsumen pesan asinkron.

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

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

/**
 * An example class to demonstrate the behavior of CLIENT_ACKNOWLEDGE mode for received
 * messages. This example
 * complements the example given in {@link SyncMessageReceiverUnorderedAcknowledge} for
 * UNORDERED_ACKNOWLEDGE mode.
 *
 * First, a session, a message producer, and a message consumer are created. Then, two
 * messages are sent. Next, two messages
 * are received but only the second one is acknowledged. After waiting for the
 * visibility time out period, an attempt to
 * receive another message is made. It's shown that no message is returned for this
 * attempt since in CLIENT_ACKNOWLEDGE mode,
 * as expected, all the messages prior to the acknowledged messages are also
 * acknowledged.
 */
```

```
* This ISN'T the behavior for UNORDERED_ACKNOWLEDGE mode. Please see {@link
SyncMessageReceiverUnorderedAcknowledge}
* for an example.
*/
public class SyncMessageReceiverClientAcknowledge {

    // Visibility time-out for the queue. It must match to the one set for the queue
    for this example to work.
    private static final long TIME_OUT_SECONDS = 1;

    public static void main(String args[]) throws JMSEException, 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 JMSEException
 */
private static void sendMessage(MessageProducer producer, Session session, String
messageText) throws JMSEException {
    // 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 JMSEException
 */
private static void receiveMessage(MessageConsumer consumer, boolean acknowledge)
throws JMSEException {
    // 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 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 * https://aws.amazon.com/apache2.0

```

```
*
* or in the "license" file accompanying this file. This file is distributed
* on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
* express or implied. See the License for the specific language governing
* permissions and limitations under the License.
*
*/

/**
 * An example class to demonstrate the behavior of UNORDERED_ACKNOWLEDGE mode for
 * received messages. This example
 * complements the example given in {@link SyncMessageReceiverClientAcknowledge} for
 * CLIENT_ACKNOWLEDGE mode.
 *
 * First, a session, a message producer, and a message consumer are created. Then, two
 * messages are sent. Next, two messages
 * are received but only the second one is acknowledged. After waiting for the
 * visibility time out period, an attempt to
 * receive another message is made. It's shown that the first message received in the
 * prior attempt is returned again
 * for the second attempt. In UNORDERED_ACKNOWLEDGE mode, all the messages must be
 * explicitly acknowledged no matter what
 * the order they're received.
 *
 * This ISN'T the behavior for CLIENT_ACKNOWLEDGE mode. Please see {@link
 * SyncMessageReceiverClientAcknowledge}
 * for an example.
 */
public class SyncMessageReceiverUnorderedAcknowledge {

    // Visibility time-out for the queue. It must match to the one set for the queue
    // for this example to work.
    private static final long TIME_OUT_SECONDS = 1;

    public static void main(String args[]) throws JMSEException, 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 JMSEException
     */
    private static void sendMessage(MessageProducer producer, Session session, String
messageText) throws JMSEException {
        // 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 JMSEException
     */
    private static void receiveMessage(MessageConsumer consumer, boolean acknowledge)
throws JMSEException {
        // 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 untuk [SpringExample.java](#).

```
<!--
    Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.

    Licensed under the Apache License, Version 2.0 (the "License").
    You may not use this file except in compliance with the License.
    A copy of the License is located at

    https://aws.amazon.com/apache2.0

    or in the "license" file accompanying this file. This file is distributed
    on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
    express or implied. See the License for the specific language governing
    permissions and limitations under the License.
-->

<?xml version="1.0" encoding="UTF-8"?>
<beans
    xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:util="http://www.springframework.org/schema/util"
    xmlns:p="http://www.springframework.org/schema/p"
    xsi:schemaLocation="
        http://www.springframework.org/schema/beans http://www.springframework.org/
schema/beans/spring-beans-3.0.xsd
        http://www.springframework.org/schema/util http://www.springframework.org/
schema/util/spring-util-3.0.xsd
    ">
```

```
<bean id="CredentialsProviderBean"
class="com.amazonaws.auth.DefaultAWSCredentialsProviderChain"/>

<bean id="ClientBuilder" class="com.amazonaws.services.sqs.AmazonSQSClientBuilder"
factory-method="standard">
    <property name="region" value="us-east-2"/>
    <property name="credentials" ref="CredentialsProviderBean"/>
</bean>

<bean id="ProviderConfiguration"
class="com.amazon.sqs.javamessaging.ProviderConfiguration">
    <property name="numberOfMessagesToPrefetch" value="5"/>
</bean>

<bean id="ConnectionFactory"
class="com.amazon.sqs.javamessaging.SQSConnectionFactory">
    <constructor-arg ref="ProviderConfiguration" />
    <constructor-arg ref="ClientBuilder" />
</bean>

<bean id="Connection" class="javax.jms.Connection"
    factory-bean="ConnectionFactory"
    factory-method="createConnection"
    init-method="start"
    destroy-method="close" />

<bean id="QueueName" class="java.lang.String">
    <constructor-arg value="SQSJMSClientExampleQueue"/>
</bean>
</beans>
```

SpringExample.java

Contoh kode Java berikut menggunakan file konfigurasi bean untuk menginisialisasi objek Anda.

```
/*
 * Copyright 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 * https://aws.amazon.com/apache2.0
```

```
*
* or in the "license" file accompanying this file. This file is distributed
* on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
* express or implied. See the License for the specific language governing
* permissions and limitations under the License.
*
*/

public class SpringExample {
    public static void main(String args[]) throws JMSEException {
        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 (JMSEException e) {
        System.err.println( "Error receiving from SQS: " + e.getMessage() );
        e.printStackTrace();
    }
}
}
```


ExampleCommon.java

Contoh kode Java berikut memeriksa apakah antrian Amazon SQS ada dan kemudian membuatnya jika tidak. Ini juga termasuk contoh kode logging.

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

public class ExampleCommon {
    /**
     * A utility function to check the queue exists and create it if needed. For most
     * use cases this is usually done by an administrator before the application is
     run.
     */
    public static void ensureQueueExists(SQSConnection connection, String queueName)
    throws JMSEException {
        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 JMSEException {
    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() );
    }
}
}
```

Didukung implementasi JMS 1.1

Perpustakaan Pesan Java Amazon SQS mendukung hal berikut [JMS 1.1 implementasi](#). 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

Tipe pesan yang didukung

- ByteMessage
- ObjectMessage
- TextMessage

Mode pengakuan pesan yang didukung

- AUTO_ACKNOWLEDGE
- CLIENT_ACKNOWLEDGE
- DUPS_OK_ACKNOWLEDGE
- UNORDERED_ACKNOWLEDGE

Note

Parameter `UNORDERED_ACKNOWLEDGE` mode bukan bagian dari spesifikasi JMS 1.1. Mode ini membantu Amazon SQS memungkinkan klien JMS untuk secara eksplisit mengakui pesan.

JMS-didefinisikan header dan properti reserved

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 \(\) AWS CloudFormation](#)
- [Tutorial: Mengirim pesan ke antrian Amazon SQS dari Amazon Virtual Private Cloud](#)

Membuat antrian Amazon SQS () AWS CloudFormation

Anda dapat menggunakan AWS CloudFormation konsol dan template JSON (atau YAKL) untuk membuat antrian Amazon SQS. Untuk informasi selengkapnya, lihat [Bekerja dengan AWS CloudFormation Template](#) dan [AWS::SQS::QueueSumber Daya](#) di Panduan AWS CloudFormation Pengguna.

Untuk digunakan AWS CloudFormation untuk membuat antrian Amazon SQS.

1. Salin kode JSON berikut ke file bernama `MyQueue.json`. Untuk membuat antrian standar, hilangkan properti `FifoQueueContentBasedDeduplication`. Untuk informasi lebih lanjut tentang deduplikasi berbasis konten, lihat [Persis-sekali diproses](#)

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 CloudFormationkonsol](#), lalu pilih Create Stack.
3. Pada panel Tentukan Template, pilih Unggah file template, pilih MyQueue . json file Anda, lalu pilih Berikutnya.
4. Pada halaman Specify Details, ketik MyQueue untuk Stack Name, lalu pilih Next.
5. Pada halaman Opsi, pilih Selanjutnya.
6. Pada halaman Tinjau, pilih Buat.

AWS CloudFormation mulai untuk membuat tumpukan MyQueue dan menampilkan status CREATE_IN_PROGRESS. Saat proses selesai, AWS CloudFormation menampilkan status CREATE_COMPLETE.

Stack Name	Created Time	Status	Description
<input checked="" type="checkbox"/> MyQueue	2017-02-20 11:39:47 UTC-0800	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 `queue.amazonaws.com` atau `us-east-2.queue.amazonaws.com`

Topik

- [Langkah 1: Buat pasangan kunci Amazon EC2](#)
- [Langkah 2: Buat AWS sumber daya](#)
- [Langkah 3: Konfirmasikan bahwa instans EC2 Anda tidak dapat diakses publik](#)
- [Langkah 4: Buat titik akhir Amazon VPC untuk Amazon SQS](#)
- [Langkah 5: Kirim pesan ke antrian Amazon SQS Anda](#)

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
- Antrian Amazon SQS

1. Unduh AWS CloudFormation template bernama [SQS-VPCE-Tutorial-CloudFormation.yaml](#) dari GitHub.
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 Berikutnya.
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-timur-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 yang 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 \(konsol\)](#)

Untuk informasi tentang menghapus sumber daya Anda, lihat berikut ini:

- [Menghapus Titik Akhir VPC](#) di Panduan Pengguna Amazon VPC
- [Hapus antrian](#)
- [Menghentikan Instans Anda](#) di Panduan Pengguna Amazon EC2 untuk Instans Linux
- [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 untuk Instans Linux

Mengotomatisasi dan pemecahan masalah antrean Amazon SQS

Bagian ini menyediakan informasi tentang mengotomatisasi dan pemecahan masalah antrean Amazon SQS.

Topik

- [Mengotomatisasi pemberitahuan dari AWS layanan ke Amazon SQS menggunakan Amazon EventBridge](#)
- [Antrean Amazon Simple Queue Service menggunakan AWS X-Ray](#)

Mengotomatisasi pemberitahuan dari AWS layanan ke Amazon SQS menggunakan Amazon EventBridge

Amazon EventBridge memungkinkan Anda mengotomatisasi AWS layanan dan merespons peristiwa sistem seperti masalah ketersediaan aplikasi atau perubahan sumber daya. Peristiwa dari AWS layanan dikirimkan ke EventBridge hampir waktu nyata. 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 mengatur berbagai sasaran—seperti standar Amazon SQS dan antrian FIFO — yang menerima peristiwa dalam format JSON. Untuk informasi selengkapnya, lihat [Amazon EventBridge sasaran](#) di dalam [Amazon EventBridge Panduan Pengguna](#).

Antrean Amazon Simple Queue Service menggunakan AWS X-Ray

AWS X-Ray mengumpulkan data tentang permintaan yang dilayani aplikasi Anda dan memungkinkan Anda melihat dan memfilter data untuk mengidentifikasi potensi masalah dan peluang pengoptimalan. Untuk setiap permintaan yang dilacak ke aplikasi Anda, Anda dapat melihat informasi yang mendetail tentang permintaan, respons, dan panggilan yang dibuat aplikasi Anda untuk membawa sumber daya AWS, microservice, database dan API web HTTP ke hilir.

Untuk mengirim AWS X-Ray header penelusuran melalui Amazon SQS, Anda dapat melakukan salah satu dari yang berikut ini:

- Menggunakan `X-Amzn-Trace-Id` [Header penelusuran](#).
- Menggunakan `AWSTraceHeader` [Atribut sistem pesan](#).

Untuk mengumpulkan data tentang kesalahan dan latensi, Anda harus instrumen [AmazonSQS](#) klien menggunakan [AWSX-Ray SDK](#).

Anda dapat menggunakan [AWS X-Ray](#) 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 [SQS dan Amazon SQS dan AWS X-Ray](#) di [AWS X-Ray Panduan Pengembang](#).

Keamanan di Amazon SQS

Bagian ini menyediakan informasi tentang keamanan, autentikasi dan kontrol akses Amazon SQS, serta Bahasa Kebijakan Akses Amazon SQS.

Topik

- [Perlindungan data](#)
- [Manajemen identitas dan akses di Amazon SQS](#)
- [Pencatatan dan pemantauan di Amazon SQS](#)
- [Validasi kepatuhan untuk Amazon SQS](#)
- [Ketahanan di Amazon SQS](#)
- [Keamanan infrastruktur di Amazon SQS](#)
- [Praktik terbaik keamanan Amazon SQS](#)

Perlindungan data

[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 memelihara kendali atas isi yang dihost 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 kredensial 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](#)
- [Privasi lalu lintas jaringan internet](#)

Enkripsi data

Perlindungan data mengacu pada perlindungan 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 diam](#)
- [Manajemen kunci](#)

Enkripsi diam

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 \(SSE\)](#)
- [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 [Konfigurasi 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 keys](#) 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

Menkripsi 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 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 antrian 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 [KeyId](#) di 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 terintegrasi dengan AWS Key Management Service (KMS) untuk mengelola [kunci](#) KMS untuk enkripsi sisi server (SSE). Lihat [Enkripsi diam](#) 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

Konfigurasi 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.

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. [Konfigurasi antrian SSE yang ada](#) menggunakan ARN kunci KMS Anda.
4. Berikan ARN dari antrian terenkripsi ke sumber acara.

Konfigurasi izin KMS untuk produsen

Ketika [periode penggunaan kembali kunci data](#) berakhir, panggilan berikutnya produsen ke `SendMessage` atau `SendMessageBatch` juga memicu panggilan ke `kms:GenerateDataKey` dan `kms:Decrypt`. Panggilan ke `kms:Decrypt` adalah 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"
  }
]}
```

```
}
```

Konfigurasi izin KMS untuk konsumen

Ketika periode penggunaan kembali kunci data berakhir, panggilan konsumen berikutnya `ReceiveMessage` juga memicu panggilan `kms: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 kunci KMS apa pun 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"
  }]
}
```

Konfigurasi izin KMS 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). Dengan demikian, 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 sebenarnya, 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 Dekripsi kesalahan](#)
- [AWS KMS GenerateDataKey kesalahan](#)

Privasi lalu lintas jaringan internet

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. Lihat informasi yang lebih lengkap di [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 sumber daya AWS secara aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diotorisasi (memiliki izin) untuk menggunakan sumber daya Amazon SQS. IAM adalah layanan Layanan AWS yang dapat Anda gunakan tanpa dikenakan 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 Anda 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

Autentikasi adalah cara Anda untuk masuk ke AWS menggunakan kredensial identitas Anda. Anda harus terautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengambil peran IAM.

Anda dapat masuk ke AWS sebagai identitas terfederasi dengan menggunakan kredensial yang disediakan melalui sumber identitas. Pengguna AWS IAM Identity Center Pengguna (Pusat Identitas IAM), autentikasi Single Sign-On perusahaan Anda, dan kredensial Google atau Facebook Anda merupakan contoh identitas terfederasi. Saat Anda masuk sebagai identitas gabungan, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil suatu peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal akses AWS. Untuk informasi selengkapnya tentang cara masuk ke AWS, lihat [Cara masuk ke Akun AWS](#) dalam Panduan Pengguna AWS Sign-In.

Jika Anda mengakses AWS secara terprogram, AWS memberikan Kit Pengembangan Perangkat Lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensial Anda. Jika Anda tidak menggunakan peralatan AWS, Anda harus menandatangani permintaan sendiri. Untuk informasi selengkapnya tentang cara menggunakan metode yang disarankan untuk menandatangani permintaan sendiri, lihat [Menandatangani permintaan API AWS](#) dalam Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Sebagai contoh, AWS menyarankan Anda menggunakan autentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari lebih lanjut, lihat

[Autentikasi multi-faktor](#) dalam Panduan Pengguna AWS IAM Identity Center dan [Menggunakan autentikasi multi-faktor \(MFA\) di AWS](#) dalam Panduan Pengguna IAM.

Pengguna root Akun AWS

Ketika membuat Akun AWS, Anda memulai dengan satu identitas masuk yang memiliki akses penuh ke semua Layanan AWS dan sumber daya di akun tersebut. Identitas ini disebut pengguna root Akun AWS dan diakses dengan cara masuk menggunakan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari Anda. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar tugas lengkap yang mengharuskan Anda masuk sebagai pengguna root, lihat [Tugas yang memerlukan kredensial pengguna root](#) dalam Panduan Pengguna IAM.

Identitas terfederasi

Praktik terbaiknya adalah mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensial temporer.

Identitas terfederasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, AWS Directory Service, direktori Pusat Identitas, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensial yang disediakan melalui sumber identitas. Ketika identitas terfederasi mengakses Akun AWS, identitas tersebut mengambil peran, dan peran ini memberikan kredensial sementara.

Untuk pengelolaan akses terpusat, sebaiknya 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 Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, lihat [Apa yang dimaksud Pusat Identitas IAM?](#) dalam Panduan Pengguna AWS IAM Identity Center.

Pengguna dan grup IAM

[Pengguna IAM](#) adalah identitas dalam Akun AWS Anda yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, sebaiknya andalkan kredensial temporer, dan bukan membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan khusus yang memerlukan kredensial jangka panjang dengan pengguna IAM, sebaiknya rotasikan 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 kumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin untuk beberapa pengguna sekaligus. Grup membuat izin lebih mudah dikelola untuk sekelompok besar pengguna. Misalnya, Anda dapat memiliki grup yang bernama IAMAdmins dan memberikan izin kepada grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran tersebut dimaksudkan untuk dapat diambil oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari selengkapnya, silakan lihat [Kapan harus membuat pengguna IAM \(bukan peran\)](#) dalam Panduan Pengguna IAM.

Peran IAM

[Peran IAM](#) merupakan identitas dalam Akun AWS Anda yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Anda dapat mengambil peran IAM untuk sementara dalam AWS Management Console dengan [berganti peran](#). Anda dapat mengambil peran dengan cara memanggil operasi API AWS CLI atau AWS atau menggunakan URL kustom. Untuk informasi selengkapnya tentang metode untuk menggunakan peran, lihat [Menggunakan peran IAM](#) dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna gabungan – Untuk menetapkan izin ke sebuah identitas gabungan, Anda dapat membuat peran dan menentukan izin untuk peran tersebut. Saat identitas terfederasi diautentikasi, identitas tersebut dikaitkan dengan peran dan diberikan izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat [Membuat peran untuk Penyedia Identitas pihak ketiga](#) dalam Panduan Pengguna IAM. Jika Anda menggunakan Pusat Identitas IAM, Anda mengonfigurasi sekumpulan izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM mengaitkan izin yang ditetapkan ke peran dalam IAM. Untuk informasi tentang rangkaian izin, lihat [Rangkaian 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 (pengguna utama tepercaya) dengan akun berbeda untuk mengakses sumber daya yang ada

di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, pada beberapa Layanan AWS, Anda dapat menyertakan kebijakan secara langsung ke sumber daya (bukan menggunakan peran sebagai proksi). Untuk mempelajari perbedaan antara kebijakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Bagaimana peran IAM berbeda dari kebijakan berbasis sumber daya](#) dalam Panduan Pengguna IAM.

- Akses lintas layanan – Sebagian Layanan AWS menggunakan fitur di Layanan AWS lainnya. Contoh, ketika Anda melakukan panggilan dalam layanan, umumnya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Suatu layanan mungkin melakukan hal tersebut menggunakan izin pengguna utama panggilan, menggunakan peran layanan, atau peran terkait layanan.
- Sesi akses maju (FAS) – Ketika Anda menggunakan pengguna IAM atau peran IAM untuk melakukan tindakan di AWS, Anda akan dianggap sebagai seorang pengguna utama. Saat menggunakan beberapa layanan, Anda mungkin melakukan tindakan yang kemudian dilanjutkan oleh tindakan lain pada layanan yang berbeda. FAS menggunakan izin dari pengguna utama untuk memanggil Layanan AWS, yang dikombinasikan dengan Layanan AWS yang diminta untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya diajukan saat layanan menerima permintaan yang memerlukan interaksi dengan Layanan AWS lain atau sumber daya lain untuk diselesaikan. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat [Meneruskan sesi akses](#).
- Peran IAM – Peran layanan adalah [peran IAM](#) yang diambil layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Membuat peran untuk mendelegasikan izin ke Layanan AWS](#) dalam Panduan pengguna IAM.
- Peran terkait layanan – Peran terkait layanan adalah tipe peran layanan yang terkait dengan Layanan AWS. Layanan tersebut dapat mengambil peran untuk melakukan sebuah tindakan atas nama Anda. Peran terkait layanan akan muncul di Akun AWS Anda dan dimiliki oleh layanan tersebut. 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 kredensial sementara untuk aplikasi yang berjalan di instans EC2 dan mengajukan permintaan API AWS CLI atau AWS. Cara ini lebih dianjurkan daripada menyimpan kunci akses dalam instans EC2. Untuk menetapkan peran AWS ke instans EC2 dan menyediakannya bagi semua aplikasinya, Anda dapat membuat profil instans yang dilampirkan ke instans tersebut. 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 di 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 mengendalikan akses di AWS dengan membuat kebijakan dan melampirkannya ke identitas atau sumber daya AWS. Kebijakan adalah objek di AWS yang, ketika terkait dengan identitas atau sumber daya, akan menentukan izinnya. AWS mengevaluasi kebijakan-kebijakan tersebut ketika seorang pengguna utama (pengguna, pengguna root, atau sesi peran) mengajukan permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan di AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, silakan lihat [Gambaran Umum kebijakan JSON](#) dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan JSON AWS untuk menentukan secara spesifik siapa yang memiliki akses terhadap apa. Artinya, pengguna utama manakah yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan pada sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat menjalankan peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk operasi. Sebagai contoh, anggap saja Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut dapat memperoleh informasi peran dari AWS Management Console, AWS CLI, atau API AWS.

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 pengguna dan peran, di sumber daya mana, dan dengan ketentuan 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 terkelola. 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 di Akun AWS Anda. Kebijakan terkelola meliputi kebijakan yang dikelola AWS dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan terkelola atau kebijakan inline, lihat [Memilih antara kebijakan terkelola 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 yang dilampiri kebijakan tersebut, kebijakan ini menentukan jenis tindakan yang dapat dilakukan oleh pengguna utama tertentu di sumber daya tersebut dan apa ketentuannya. Anda harus [menentukan pengguna utama](#) dalam kebijakan berbasis sumber daya. Pengguna utama dapat mencakup akun, pengguna, peran, pengguna gabungan, atau Layanan AWS.

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan yang dikelola AWS dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACL)

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 tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, silakan lihat [Gambaran umum daftar kontrol akses \(ACL\)](#) di Panduan Developer Layanan Penyimpanan Ringkas Amazon.

Tipe kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Tipe-tipe kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda berdasarkan tipe kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan di mana Anda menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas IAM (pengguna 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 secara eksplisit terhadap salah satu kebijakan ini akan mengesampingkan izin tersebut. 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 sebuah organisasi atau unit organisasi (OU) di AWS Organizations. AWS Organizations adalah layanan untuk mengelompokkan dan mengelola beberapa akun AWS yang dimiliki bisnis Anda secara terpusat. 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 dalam akun anggota, termasuk setiap 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 teruskan sebagai parameter saat Anda membuat sesi sementara secara terprogram untuk peran atau pengguna gabungan. 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 di salah satu kebijakan ini akan membatalkan izin tersebut. Untuk informasi selengkapnya, lihat [Kebijakan sesi](#) dalam Panduan Pengguna IAM.

Berbagai jenis kebijakan

Jika beberapa jenis kebijakan diberlakukan untuk satu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan ketika ada beberapa jenis kebijakan, lihat [Logika evaluasi kebijakan](#) dalam Panduan Pengguna IAM.

Ikhtisar mengelola akses di Amazon SQS

Setiap sumber daya AWS 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 tentang administrator, 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
Antrian	<code>arn:aws:sqs: <i>region</i>:<i>account_id</i> :<i>queue_name</i></code>

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 dalam akun, terlepas dari siapa yang membuat sumber daya tersebut. 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 antrian 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 antrian. Namun, Anda Akun AWS (milik pengguna) memiliki sumber daya antrian.
- Jika Anda membuat peran IAM Akun AWS dengan izin untuk membuat antrian Amazon SQS, siapa pun yang dapat mengambil peran tersebut dapat membuat antrian. 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 (kebijakan IAM dan kebijakan Amazon SQS)


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 antrian Amazon SQS, lampirkan kebijakan izin ke pengguna atau grup tempat pengguna tersebut berada.
- Lampirkan kebijakan izin ke pengguna lain Akun AWS — Untuk memberikan izin pengguna untuk membuat antrian 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.

SQS Policy on `queue_xyz`

Allow who:

AWS account 1
AWS account 2

Actions: *

Resource:

`123456789012/queue_xyz`

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.

SQS Policy on `queue_xyz`

Allow who:

AWS account 1
AWS account 2

User Bob
User Susan (in your own account)

Actions: *

Resource:

`arn:aws:sqs:*:123456789012/queue_xyz`

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

tindakan yang termasuk dalam * telah diperluas. Untuk daftar tindakan yang diizinkan, lihat [izin Amazon SQS API: Tindakan dan referensi sumber daya](#).

3

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 ID123456789012) 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, lihat [Menggunakan 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—baik mengizinkan maupun menolak. 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, lihat [Izin 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, [AWSlihat layanan yang bekerja dengan IAM di Panduan Pengguna IAM](#).

Kontrol akses

Daftar kontrol akses (ACL) mengontrol pengguna utama (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, silakan lihat [Gambaran umum daftar kontrol akses \(ACL\)](#) di Panduan Developer Layanan Penyimpanan Ringkas Amazon.

Note

Penting untuk memahami bahwa semua Akun AWS dapat mendelegasikan izinnya kepada pengguna di bawah akunnya. Akses lintas akun memungkinkan Anda berbagi akses ke sumber daya AWS 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](#) 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 pengguna dan peran, di sumber daya mana, dan dengan ketentuan apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Membuat kebijakan IAM](#) dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan tindakan dan sumber daya yang diizinkan atau ditolak, serta ketentuan terkait jenis tindakan yang diizinkan atau ditolak. Anda tidak dapat menentukan pengguna utama dalam kebijakan berbasis identitas karena kebijakan ini berlaku untuk pengguna atau peran yang dilampiri kebijakan. Untuk mempelajari semua elemen yang dapat digunakan 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 yang dilampiri kebijakan tersebut, kebijakan ini menentukan jenis tindakan yang dapat dilakukan oleh pengguna utama tertentu di sumber daya tersebut dan apa ketentuannya. Anda harus [menentukan pengguna utama](#) dalam kebijakan berbasis sumber daya. Pengguna utama dapat mencakup akun, pengguna, peran, pengguna gabungan, atau Layanan AWS.

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan seluruh akun atau entitas IAM di akun lain sebagai pengguna utama dalam kebijakan berbasis sumber daya. Menambahkan pengguna utama lintas akun ke kebijakan berbasis sumber daya bagian dari membangun hubungan kepercayaan. Ketika pengguna utama dan sumber daya berada di Akun AWS yang berbeda, administrator IAM di akun tepercaya juga harus memberikan izin kepada entitas pengguna utama (pengguna atau peran) untuk mengakses sumber daya. Izin diberikan dengan melampirkan kebijakan berbasis identitas ke entitas tersebut. Namun, jika kebijakan berbasis sumber daya memberikan akses kepada pengguna utama dalam akun yang sama, kebijakan berbasis identitas lainnya tidak diperlukan. Untuk informasi selengkapnya, lihat [Perbedaan peran IAM dengan kebijakan berbasis sumber daya](#) di Panduan Pengguna IAM.

Tindakan kebijakan untuk Amazon SQS

Mendukung tindakan kebijakan Ya

Administrator dapat menggunakan kebijakan JSON AWS untuk menentukan siapa yang memiliki akses ke apa. Yaitu, pengguna utama mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam 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 seperti operasi API AWS 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 suatu kebijakan untuk memberikan izin 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-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 JSON AWS untuk menentukan siapa yang memiliki akses ke apa. Yaitu, pengguna utama mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek atau beberapa 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 mengindikasikan 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 spesifik layanan	Ya
--	----

Administrator dapat menggunakan kebijakan JSON AWS untuk menentukan siapa yang memiliki akses ke apa. Yaitu, pengguna utama mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Elemen Condition (atau blok Condition) memungkinkan Anda menentukan kondisi di mana suatu pernyataan akan diterapkan. Elemen Condition bersifat opsional. Anda dapat membuat ekspresi kondisional 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 satu pernyataan, atau beberapa kunci dalam satu elemen Condition, AWS akan mengevaluasinya dengan menggunakan operasi AND logis. Jika Anda menentukan beberapa nilai untuk satu kunci persyaratan, AWS akan mengevaluasi syarat tersebut menggunakan operasi OR yang 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 tanda yang sesuai dengan nama pengguna IAM mereka. Untuk informasi selengkapnya, silakan lihat [Elemen kebijakan IAM: variabel dan tanda](#) di Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi spesifik layanan. Untuk melihat semua kunci kondisi global AWS, lihat [kunci konteks kondisi global AWS](#) dalam 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) mengontrol pengguna utama (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL sama dengan kebijakan berbasis sumber daya, meskipun 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. Di AWS, atribut ini disebut tag. Anda dapat melampirkan tanda ke entitas IAM (pengguna atau peran) dan ke banyak sumber daya AWS. Pemberian tanda ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi-operasi ketika tanda milik pengguna utama cocok dengan tanda yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna dalam situasi di mana pengelolaan kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tag, berikan informasi tentang tanda di [elemen syarat](#) dari sebuah kebijakan dengan menggunakan kunci-kunci persyaratan `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 hanya untuk beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya tentang ABAC, lihat [Apa itu ABAC?](#) di Panduan Pengguna IAM. Untuk melihat tutorial terkait langkah-langkah penyiapan ABAC, lihat [Menggunakan kontrol akses berbasis atribut \(ABAC\)](#) di Panduan Pengguna IAM.

Menggunakan kredensial sementara dengan Amazon SQS

Mendukung kredensial sementara	Ya
--------------------------------	----

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensial sementara. Sebagai informasi tambahan, termasuk tentang Layanan AWS mana saja yang berfungsi dengan kredensial sementara, lihat [Layanan AWS yang berfungsi dengan IAM](#) di Panduan Pengguna IAM.

Anda menggunakan kredensial sementara jika Anda masuk ke AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS dengan menggunakan tautan masuk tunggal (SSO) milik perusahaan Anda, proses itu secara otomatis akan membuat kredensial temporer. Anda juga akan membuat kredensial sementara secara otomatis saat masuk ke konsol sebagai pengguna dan kemudian beralih peran. Untuk informasi selengkapnya tentang cara beralih peran, lihat [Beralih peran \(konsol\)](#) di Panduan Pengguna IAM.

Anda dapat membuat kredensial sementara secara manual menggunakan AWS CLI atau AWS API. Anda kemudian dapat menggunakan kredensial sementara untuk mengakses AWS. AWS menyarankan Anda membuat kredensial sementara secara dinamis, 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
---------------------------------	----

Jika menggunakan pengguna IAM atau peran IAM untuk melakukan tindakan di AWS, Anda akan dianggap sebagai pengguna utama. Jika menggunakan beberapa layanan, Anda mungkin melakukan tindakan yang kemudian dilanjutkan oleh tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pengguna utama untuk memanggil Layanan AWS, yang dikombinasikan dengan Layanan AWS yang diminta untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya diajukan saat layanan menerima permintaan yang memerlukan interaksi dengan Layanan AWS lain atau sumber daya lain untuk diselesaikan. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat [Meneruskan sesi akses](#).

Peran layanan untuk Amazon SQS

Mendukung peran layanan	Ya
-------------------------	----

Peran layanan adalah [peran IAM](#) yang diambil 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 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 tertaut layanan adalah jenis peran layanan yang terkait dengan Layanan AWS. Layanan ini dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan akan muncul di Akun AWS Anda dan dimiliki oleh layanan tersebut. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Untuk detail tentang pembuatan atau pengelolaan peran terkait layanan, lihat [Layanan AWS yang berfungsi dengan IAM](#). Temukan sebuah 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 lebih lanjut tentang kebijakan-kebijakan terkelola AWS, lihat [kebijakan terkelola AWS](#) di Panduan Pengguna IAM.

Layanan AWS mempertahankan dan memperbarui kebijakan-kebijakan terkelola AWS. Anda tidak dapat mengubah izin yang ada dalam kebijakan-kebijakan yang dikelola AWS. Layanan terkadang menambahkan izin tambahan ke kebijakan yang dikelola AWS untuk mendukung fitur-fitur baru. Jenis pembaruan ini akan memengaruhi semua identitas (pengguna, grup, dan peran) di mana kebijakan tersebut dilampirkan. Layanan kemungkinan besar akan memperbarui kebijakan yang dikelola AWS saat ada fitur baru yang diluncurkan atau saat ada operasi baru yang tersedia. Layanan tidak menghapus izin yang ada di kebijakan yang dikelola AWS, sehingga pembaruan-pembaruan yang terjadi pada kebijakan tidak akan membuat izin yang ada rusak.

Selain itu, AWS mendukung kebijakan-kebijakan terkelola untuk fungsi tugas yang mencakup beberapa layanan. Sebagai contoh, kebijakan ReadOnlyAccess terkelola AWS menyediakan akses hanya-baca ke semua layanan dan sumber daya AWS. Saat layanan meluncurkan fitur baru, AWS menambahkan 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.

AWSkebijakan 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

AWSkebijakan terkelola: AmazonSQS ReadOnlyAccess

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 [AmazonSQS ReadOnlyAccess](#) 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
AmazonSQS ReadOnlyAccess	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 `sqz:GetWidget` fiktif.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
sqz:GetWidget on resource: my-example-widget
```

Dalam hal ini, kebijakan Mateo harus diperbarui untuk memungkinkannya mengakses `my-example-widget` sumber daya menggunakan `sqz:GetWidget` tindakan tersebut.

Jika Anda membutuhkan bantuan, hubungi administrator AWS 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.

Sebagian Layanan AWS mengizinkan Anda untuk memberikan peran yang sudah ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait-layanan. Untuk melakukan tindakan tersebut, Anda harus memiliki izin untuk memberikan peran pada layanan tersebut.

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 membutuhkan bantuan, hubungi administrator AWS 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 pengguna 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 pengguna akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa hal berikut:

- Untuk mengetahui apakah Amazon SQS mendukung fitur-fitur ini, lihat [Bagaimana Amazon Simple Queue Service bekerja dengan IAM](#)
- Untuk mempelajari cara memberikan akses ke sumber daya di seluruh Akun AWS yang Anda miliki, lihat [Menyediakan akses ke pengguna IAM di Akun AWS lainnya yang Anda miliki](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses ke sumber daya Anda ke pihak ketiga Akun AWS, lihat [Menyediakan akses ke akun Akun AWS yang dimiliki oleh pihak ketiga](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat [Menyediakan akses ke pengguna terautentikasi eksternal \(gabungan identitas\)](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara penggunaan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Perbedaan antara peran IAM dan kebijakan berbasis sumber daya](#) 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](#).

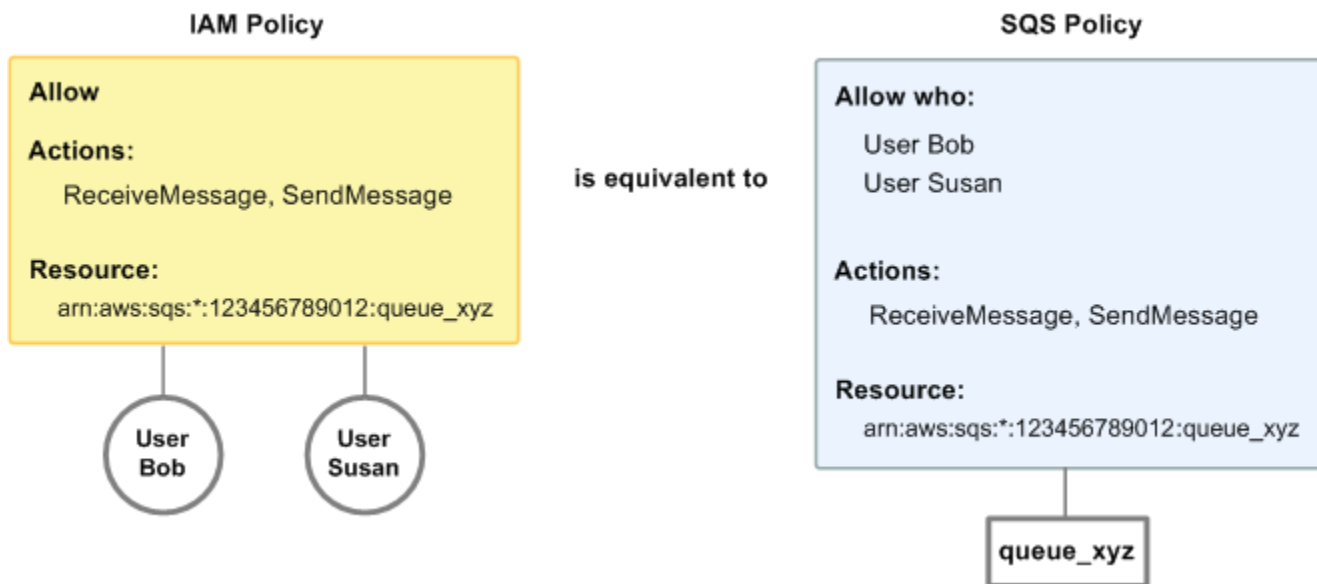
Topik

- [Menggunakan kebijakan Amazon SQS dan IAM](#)
- [Izin diperlukan untuk menggunakan konsol Amazon SQS](#)
- [Contoh kebijakan berbasis identitas untuk Amazon SQS](#)
- [Contoh dasar kebijakan Amazon SQS](#)
- [Menggunakan kebijakan khusus dengan Bahasa Kebijakan Akses Amazon SQS](#)

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 ini 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.

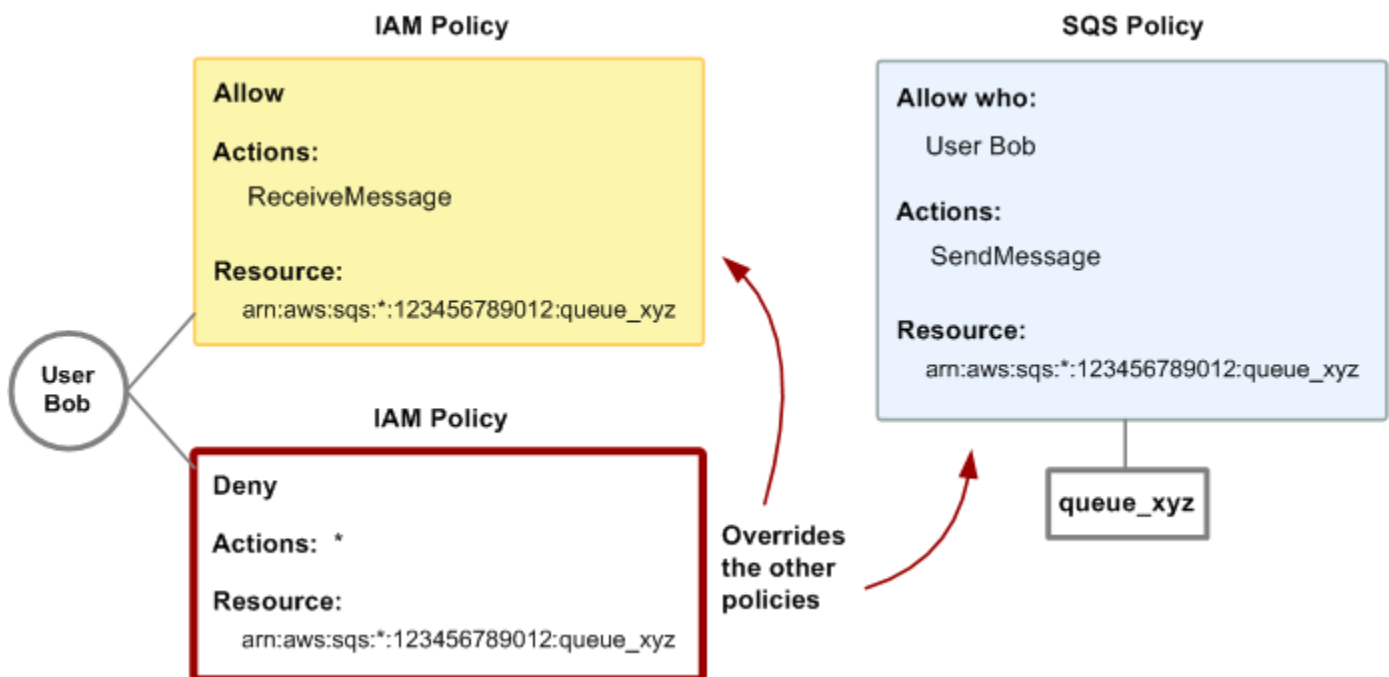
Terserah Anda bagaimana Anda menggunakan kedua sistem bersama-sama untuk mengelola izin Anda. Contoh berikut menunjukkan cara sistem dua kebijakan bekerja sama.

- 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, lihat[Menggunakan 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. Pengguna dan peran tersebut juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau API AWS. Untuk memberikan izin kepada pengguna untuk melakukan tindakan pada sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat menjalankan 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 untuk Instans Linux.

Topik

- [Praktik terbaik kebijakan](#)
- [Menggunakan konsol Amazon SQS](#)
- [Izinkan 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 dikenai biaya untuk Akun AWS Anda. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulai menggunakan kebijakan yang dikelola AWS dan beralih ke izin dengan hak akses paling rendah – Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan yang dikelola AWS yang memberikan izin untuk banyak kasus penggunaan umum. Kebijakan ini ada di Akun AWS Anda. Sebaiknya Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan AWS yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat [kebijakan yang dikelola AWS](#) atau [kebijakan yang dikelola AWS untuk fungsi pekerjaan](#) di 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 melakukan ini dengan menentukan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, juga dikenal sebagai izin hak akses paling rendah. Untuk informasi selengkapnya tentang

cara menggunakan IAM untuk menerapkan izin, lihat [Kebijakan dan izin di IAM](#) di Panduan Pengguna IAM.

- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Misalnya, Anda dapat menulis syarat kebijakan untuk menentukan bahwa semua pengajuan harus dikirim menggunakan SSL. Anda juga dapat menggunakan kondisi untuk memberi akses ke tindakan layanan jika digunakan melalui Layanan AWS yang spesifik, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat [Elemen kebijakan JSON IAM: Syarat](#) di Panduan Pengguna IAM.
- Menggunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda guna memastikan izin yang aman dan berfungsi – 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 Analizer Akses IAM](#) di Panduan Pengguna IAM.
- Wajibkan autentikasi multi-faktor (MFA) – Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Akun AWS Anda, aktifkan MFA untuk keamanan tambahan. Untuk mewajibkan MFA saat operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat [Mengonfigurasi akses API yang dilindungi MFA](#) di Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat [Praktik terbaik keamanan di IAM](#) di 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 memberikan izin konsol minimum untuk pengguna yang melakukan panggilan hanya ke AWS CLI atau API AWS. Sebaliknya, izinkan akses hanya ke tindakan yang cocok dengan operasi API yang coba dilakukan.

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 [Menambahkan izin ke pengguna](#) di Panduan Pengguna IAM.

Izinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan para pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan pada konsol atau menggunakan AWS CLI atau AWS API secara terprogram.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsWithUser",
        "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 antrian. 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, danSendMessage.`

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 izin segalanya kecuali. 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 kredensial 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 yang diberi namaWidgetPartnerQueue.

```
{
  "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 (111122223333 dan 444455556666) 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](#)

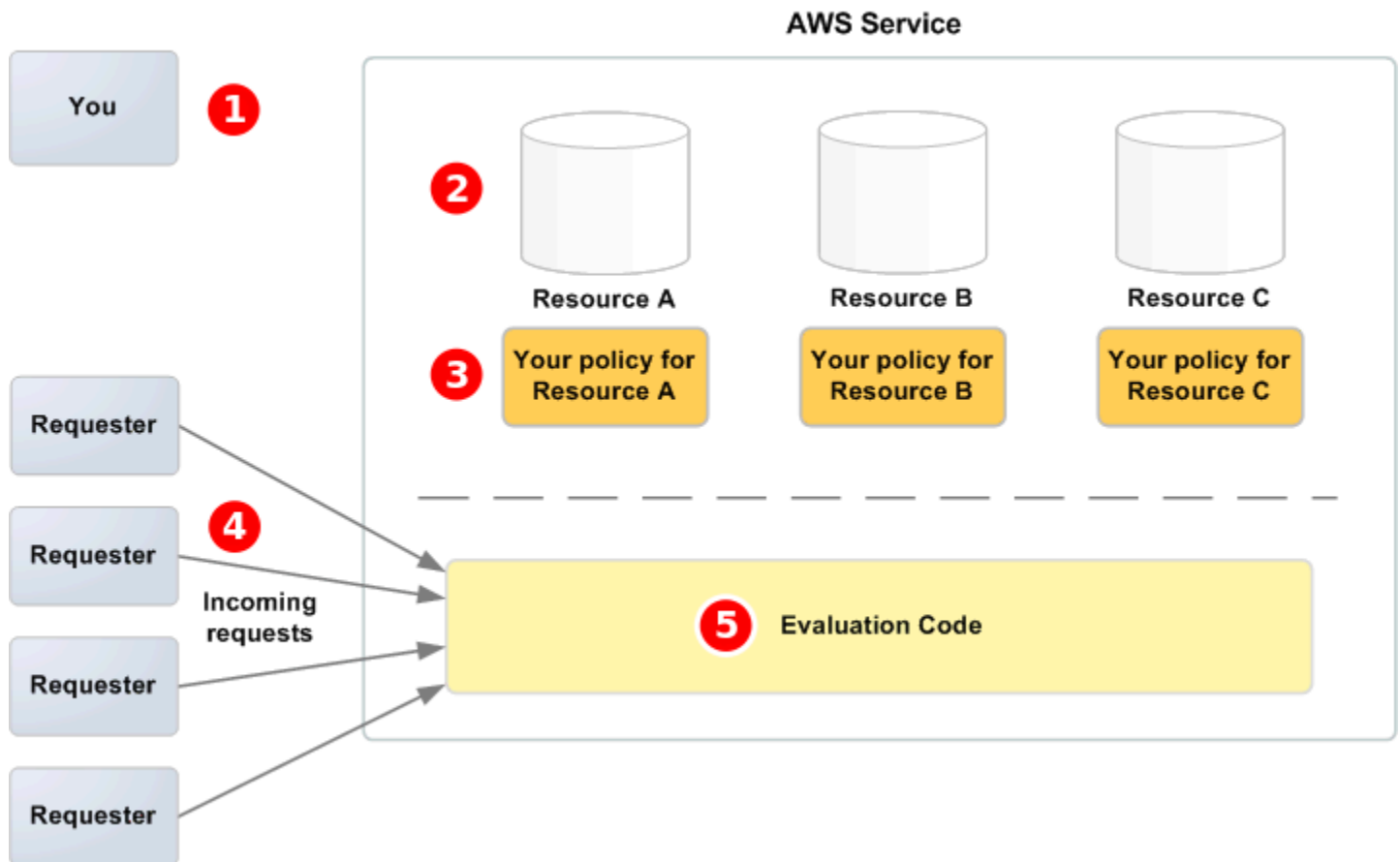
Jika Anda ingin secara eksplisit menolak atau mengizinkan akses berdasarkan kondisi yang lebih spesifik (seperti waktu permintaan masuk atau alamat IP pemohon), Anda perlu menulis kebijakan Amazon SQS Anda sendiri dan mengunggahnya ke sistem menggunakan tindakan Amazon SQS. [AWS SetQueueAttributes](#)

Topik

- [Arsitektur kontrol akses Amazon SQS](#)
- [Alur kerja proses kontrol akses Amazon SQS](#)
- [Konsep kunci bahasa Kebijakan Akses Amazon SQS](#)
- [Logika evaluasi bahasa Kebijakan Akses Amazon SQS](#)
- [Hubungan antara penolakan eksplisit dan default dalam Bahasa Kebijakan Akses Amazon SQS](#)
- [Batasan Kebijakan Kustom](#)
- [Contoh Bahasa Kebijakan Akses Amazon SQS Kustom](#)

Arsitektur kontrol akses Amazon SQS

Diagram berikut menjelaskan kontrol akses untuk sumber daya Amazon SQS Anda.



1 Anda, pemilik sumber daya.

2 Sumber daya Anda yang terkandung dalam AWS layanan (misalnya, antrian Amazon SQS).

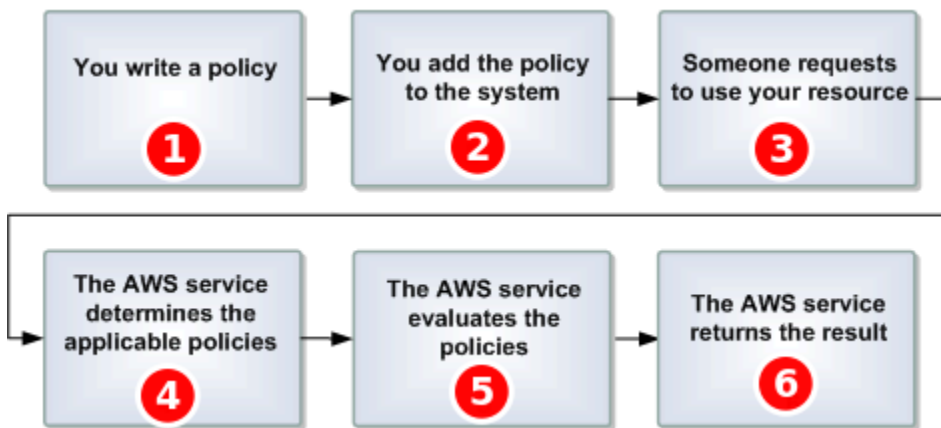
3 Kebijakan Anda. Merupakan praktik yang baik untuk memiliki satu kebijakan per sumber daya. AWS Layanan ini menyediakan API yang Anda gunakan untuk mengunggah dan mengelola kebijakan Anda.

4 Pemohon dan permintaan masuk mereka ke layanan. AWS

5 Kode evaluasi bahasa kebijakan akses. Hal ini adalah rangkaian kode dalam layanan AWS yang mengevaluasi permintaan masuk terhadap kebijakan yang dapat diterapkan dan menentukan apakah peminta diizinkan mendapat akses ke 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 Seseorang mengirim permintaan untuk menggunakan antrian Amazon SQS Anda.

4 Amazon SQS memeriksa semua kebijakan Amazon SQS yang tersedia dan menentukan kebijakan mana yang berlaku.

5 Amazon SQS mengevaluasi kebijakan dan menentukan apakah pemohon diizinkan untuk menggunakan antrian Anda.

6 Berdasarkan 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 ke `allow`.

Aksi

Aktivitas yang [Kepala Sekolah](#) memiliki izin untuk melakukan, biasanya permintaan untuk AWS.

Penyangkalan default

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: [Penyangkalan default](#), [Izinkan](#), dan [Penyangkalan eksplisit](#).

Penyangkalan eksplisit

Hasil dari a [Pernyataan](#) yang telah [Efek](#) diatur ke `deny`.

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 a [Kunci](#).

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](#) di [Kebijakan](#).

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 file [Sumber Daya](#).

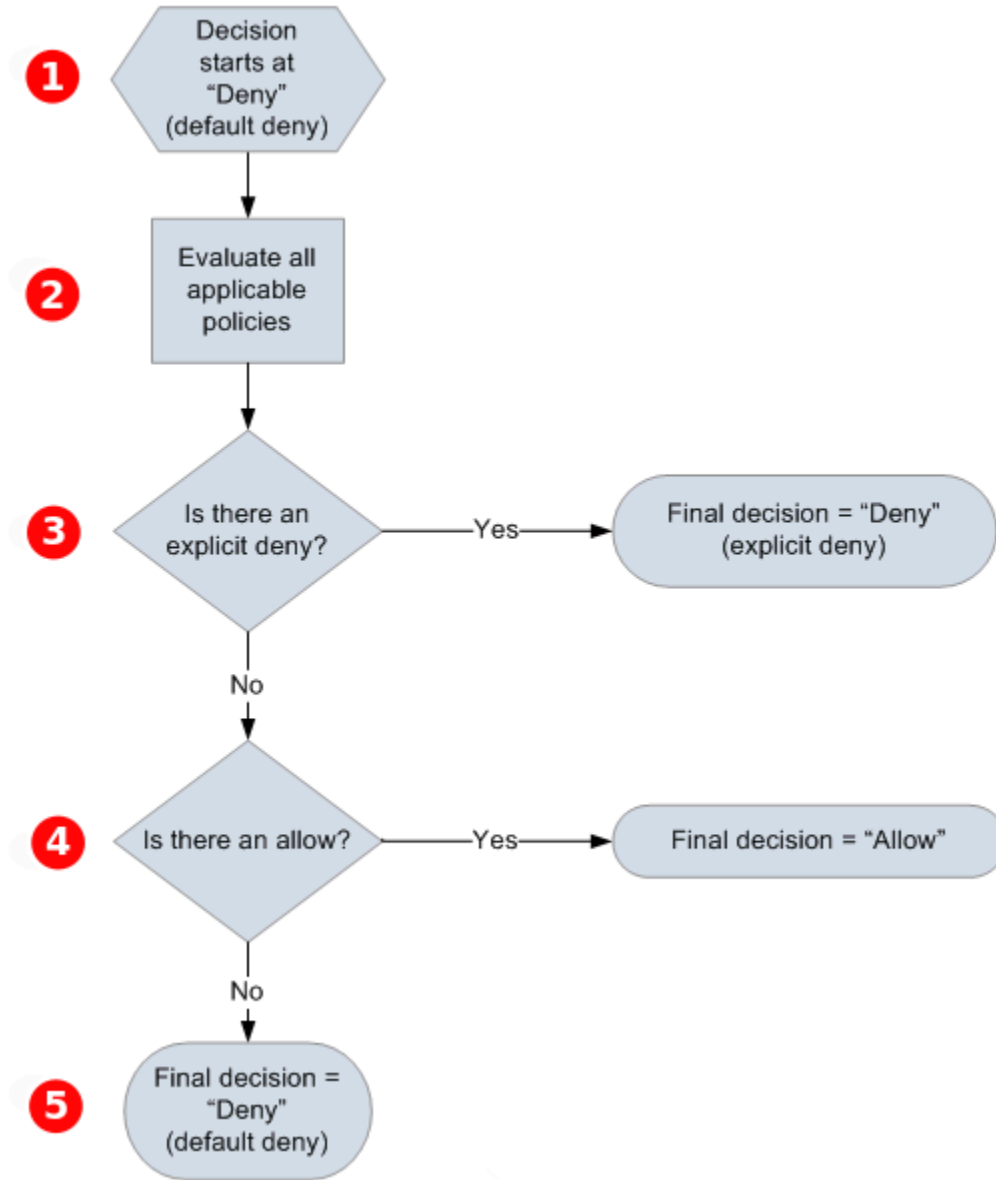
Logika evaluasi bahasa Kebijakan Akses Amazon SQS

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 [Izinkan](#) mengesampingkan apa pun. [Penyangkalan default](#)
- Sebuah [Penyangkalan eksplisit](#) mengesampingkan izin apa pun.
- Urutan di mana kebijakan dievaluasi tidak penting.

Diagram berikut menjelaskan secara rinci bagaimana Amazon SQS mengevaluasi keputusan tentang izin akses.



1 dimulai dengan default-deny.

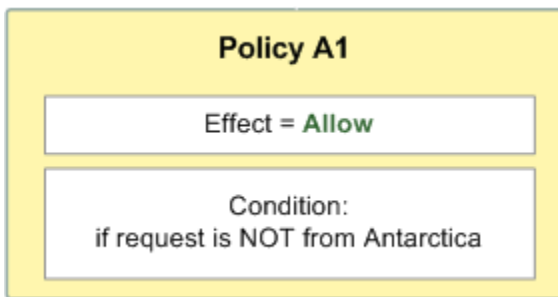
Keputu

- 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).

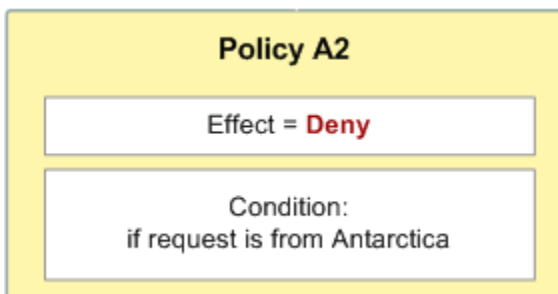
Hubungan antara penolakan eksplisit dan default dalam Bahasa Kebijakan Akses Amazon SQS

Jika kebijakan Amazon SQS tidak langsung berlaku untuk permintaan, permintaan akan menghasilkan [Penyangkalan default](#). 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 menghasilkan baik [Izinkan](#) atau [Penyangkalan eksplisit](#) berdasarkan nilai [Efek](#) elemen 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.

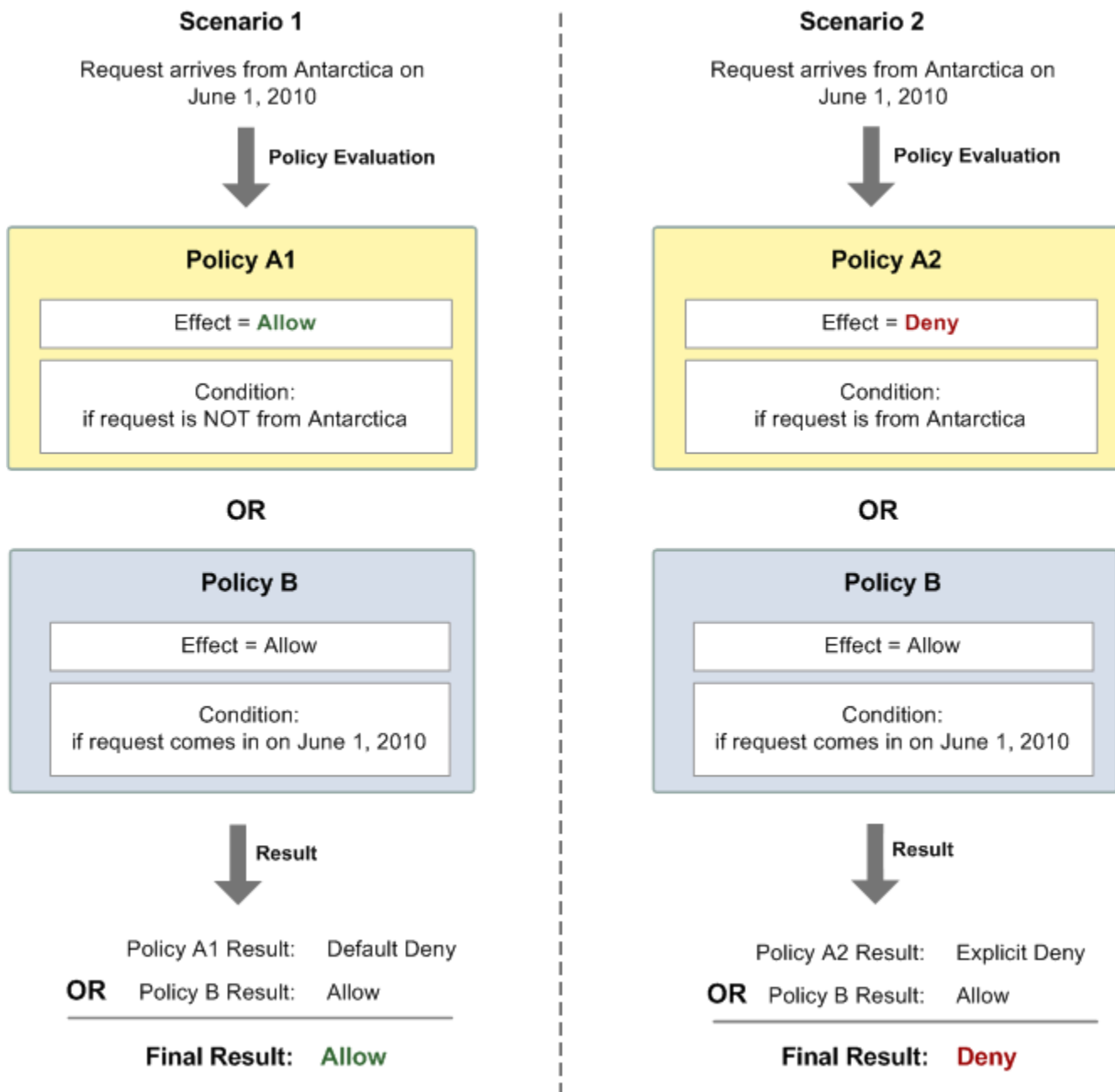


Jika pengguna mengirim permintaan dari AS, kondisi terpenuhi (permintaan bukan dari Antartika), dan permintaan tersebut menghasilkan izin. Namun, jika pengguna mengirim permintaan dari Antartika, kondisi tidak terpenuhi dan permintaan default ke default-deny. Anda dapat mengubah hasilnya menjadi penolakan 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

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 [SetQueueAttribute](#) 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 ke `queue1: 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 kredensial 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 kredensi sementara.

Prasyarat

1. Gunakan IAM untuk membuat kredensial 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 kredensial 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=wJa1rXUtnFEMI/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
```

Mengelola akses ke antrian Amazon SQS terenkripsi Anda menggunakan kebijakan dan kebijakan kunci Amazon SQS yang paling tidak memiliki hak istimewa AWS KMS

[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 antrian 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 antrian Amazon SQS dan datanya.

Bagian berikut memberikan informasi tentang cara mengontrol akses ke antrian Amazon SQS terenkripsi melalui kebijakan Amazon SQS dan kebijakan utama. AWS KMS Kebijakan dalam panduan ini akan membantu Anda mencapai [hak istimewa paling sedikit](#).

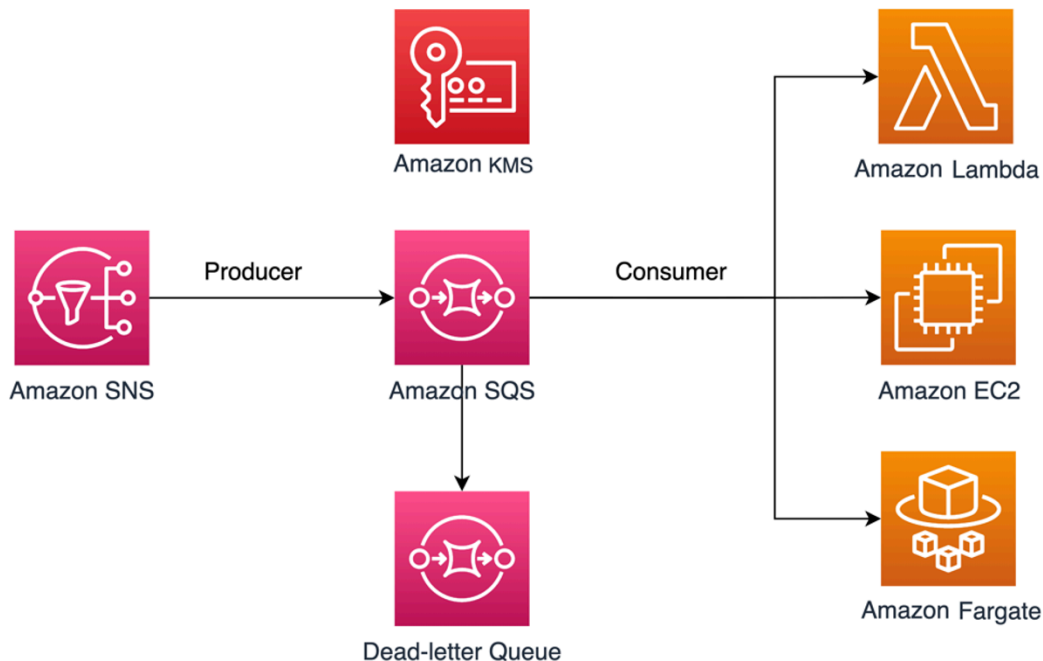
Panduan ini juga menjelaskan bagaimana kebijakan berbasis sumber daya mengatasi [masalah wakil yang membingungkan dengan menggunakan kunci konteks kondisi IAM `aws:SourceArn`, `aws:SourceAccount` dan global. `aws:PrincipalOrgID`](#)

Topik

- [Ikhtisar](#)
- [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](#)

Ikhtisar

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 antrian Amazon SQS terenkripsi Anda. Konsumen pesan adalah layanan komputasi, seperti [AWS Lambda](#) fungsi, 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 antrian Amazon SQS. Jika konsumen pesan berada di cloud pribadi virtual (VPC), pernyataan [DenyReceivingIfNotThroughVPC](#) 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 antrian 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 antrian 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 antrian 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 Anda.](#)

```
{
  "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 KMSkunci 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": "AllowReadAccesssToKeyMetaData",
  "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 mempublikasikan pesan ke antrian 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

ConditionDalam pernyataan membatasi akses hanya ke layanan Amazon SNS di akun yang samaAWS.

```

{
  "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 antrian 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 antrian, mengelola antrian, atau keduanya. Pastikan untuk mengganti <placeholder values> dengan informasi Anda sendiri. Tentukan ARN peran IAM yang digunakan untuk menerapkan antrian 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 antrian 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 antrian 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 antrian 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 antrian 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 deputy 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 pemanggil) 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:SourceArns: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 [AWSIAM 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 tersebut dimaksudkan dan aman, atau apakah akses tersebut 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 [AWSIAM 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*

ChangeMessageVisibility

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*

GetQueueAttributes

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`

[ListDeadLetterSourceQueues](#)

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`

[ListQueueTags](#)

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`

[SendMessage](#) dan [SendMessageBatch](#)

Tindakan: `sqs:SendMessage`

Sumber daya: `arn:aws:sqs:region:account_id:queue_name`

[SetQueueAttributes](#)

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 logging dan pemantauan antrian Amazon SQS.

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.

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 kejadian terbaru untuk akun AWS 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 antrian 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 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 lebih lanjut, lihat [AWS CloudTrail Harga](#).

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.amazonaws.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.amazonaws.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.amazonaws.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 untukSendMessage.

```
{
  "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": {
    "mD50fMessageBody": "9a4e3f7a614d9dd9f8722092dbda17a2",
    "mD50fMessageSystemAttributes": "f88f0587f951b7f5551f18ae699c3a9d",
  }
}
```

```

    "messageId": "93bb6e2d-1090-416c-81b0-31eb1faa8cd8",
    "sequenceNumber": "18881790870905840128"
  },
  "requestID": "c4584600-fe8a-5aa3-a5ba-1bc42f055fae",
  "eventID": "98c735d8-70e0-4644-9432-b6ced4d791b1",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::SQS::Queue",
      "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "123456789012",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
  }
}

```

SendMessageBatch

Contoh berikut menunjukkan peristiwa CloudTrail data untukSendMessageBatch.

```

{
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
      }
    }
  }
}

```



```
    },
    "attributes": {
      "creationDate": "2023-11-07T22:13:06Z",
      "mfaAuthenticated": "false"
    }
  }
},
"eventTime": "2023-11-07T23:59:05Z",
"eventSource": "sqs.amazonaws.com",
"eventName": "SendMessageBatch",
"awsRegion": "ap-southeast-4",
"sourceIPAddress": "10.0.118.80",
"userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
"requestParameters": {
  "delaySeconds": 0,
  "entries": [
    {
      "id": "0",
      "messageBody": "HIDDEN_DUE_TO_SECURITY_REASONS",
      "messageAttributes": [
        {
          "name": "HIDDEN_DUE_TO_SECURITY_REASONS"
        }
      ],
      "messageDeduplicationId": "MsgDedupIdSdk1027092b6-b6f6-41af-a084-e72d572a6d4b",
      "messageGroupId": "MsgGroupIdSdk12"
    }
  ],
  "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue"
},
"responseElements": {
  "successful": [
    {
      "id": "0",
      "messageId": "9048ab28-e38d-46da-b9fe-f70b3873f888",
      "mD50fMessageBody": "0f1a575a56eb5cf5072a8dedc585d2dd",
      "mD50fMessageAttributes": "6e1d6d5d774a05efe9df5eb000639db7",
      "sequenceNumber": "18881790869375471872",
      "mD50fMessageSystemAttributes": "6f540b6e375dcda1aad2d4aaff28ebf8"
    }
  ]
},
},
```

```

"requestID": "b5a386a4-2d4a-5de3-9910-db60fcc368ee",
"eventID": "20f5ecbe-2b0b-4d0b-a6f7-365bc94c4ca5",
"readOnly": false,
"resources": [
  {
    "accountId": "123456789012",
    "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue",
    "type": "AWS::SQS::Queue"
  }
],
"eventType": "AwsApiCall",
"managementEvent": false,
"recipientAccountId": "123456789012",
"eventCategory": "Data",
"tlsDetails": {
  "tlsVersion": "TLSv1.2",
  "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
  "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
}
}

```

ReceiveMessage

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"
}
}
```

DeleteMessage

Contoh berikut menunjukkan peristiwa CloudTrail data untuk DeleteMessage.

```

{
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
      },
      "attributes": {
        "creationDate": "2023-11-07T22:13:06Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:58:42Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "DeleteMessage",
  "awsRegion": "ap-southeast-4",
  "sourceIPAddress": "10.0.118.80",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "receiptHandle": "AQEBfgYUKTy3dy0AewC4wI3lQZEB3oUDuv8M8FWh0bnr3lqRsFBiZ57mmx01/dWfdlvGgDW7sRSry6HHxWrNfHItQMUHtWX3a/vEjJ6sWC/5Mf36I/B2HBLCT2zG0/IZTywxFmUT4HUudWkCgpuZb/Kcl3Fom6hYU8PxxzPxL0KPtFwrVU+G2Spvf/Tbuyj27h5+AkNxfAhu/dnvXnAJcDJErgsJTjSS1i6iRzFq+jg6K5Fw6T578QJZcx/ZLaCyohmj2Ha00ktwhbqQc4j+2gKSfxrACgXCu6De5bCtwgtGdhMEh4DtVIQh88qGUcaofQ3t/eRBIvIFJIA61JCVNWSBq0tELEIfxaHpSvo0c1IEeckDt1IJ08Cij3euLFMIzmUot24IViZt8ntKVAZ6KBL1LedrV1x0hNVcWG0jfbqz3iBS1T1AD1zJKT7ICIA+edgaYJp0Zw4=",
    "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue"
  },
  "responseElements": null,
  "requestID": "fbd23ff4-a107-536d-8fcb-623070754bc0",

```

```
"eventID": "9951fed7-365f-4046-bc71-e5bf065a9b47",
"readOnly": false,
"resources": [
  {
    "accountId": "123456789012",
    "type": "AWS::SQS::Queue",
    "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
  }
],
"eventType": "AwsApiCall",
"managementEvent": false,
"recipientAccountId": "123456789012",
"eventCategory": "Data",
"tlsDetails": {
  "tlsVersion": "TLSv1.2",
  "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
  "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
}
}
```

DeleteMessageBatch

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": "AQEBefxM104zyZGF87DehbRbmri91w2W7mMdD0GrBjQa8e/
hpb4RbXHPZ9tLBV1eECbChQIE5NtaDuoZhZP0kTy0eN46EyRR4jXDzE3A1kbP1X1mA9f2fUuTrXx8aeCoCA3I3woNg3fXXA
hLLS94tjAZqV2krc4BaC2pYgjyHwCw019HwIV8T/bjNMIEZoQwOM5V
+o9vHPfewz5QGr5SKpDo7uE7Umyk5n5CJZvcn1efp/
mrwtaCIb9M7cCQUYcZm2ZmZDnI09XpGTai3m2dQ0M83pnNh0nvDfPkHpoa+hX1TrUmxCupCWHJwA8HFJ10/
CCJsodMNFthLBA9S57dkBZCsw41G8jAmgQ0MkvZ0UL5mg00FQqd1Yrw0zvtHjCgiwdzn0yXoMzxIZMBxkY14E4nVVZ7N5XE
h8oRk2C7gByzg2kYJ0LnUvLJFT8DQE28JZppEC9klvrdR/BWiPT7asc="
    }
  ]
},
"responseElements": {
  "successful": [
    {
      "id": "0"
    }
  ],
  "failed": []
},
"requestID": "fe423091-5642-5ba5-9256-6d5587de52f1",
"eventID": "88c8020d-d769-4985-8ecb-ee0b59acc418",
"readOnly": false,
"resources": [
  {
    "accountId": "123456789012",
    "type": "AWS::SQS::Queue",
    "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
  }
]
}

```

```
],
"eventType": "AwsApiCall",
"managementEvent": false,
"recipientAccountId": "123456789012",
"eventCategory": "Data",
"tlsDetails": {
  "tlsVersion": "TLSv1.2",
  "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
  "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
}
}
```

ChangeMessageVisibility

Contoh berikut menunjukkan peristiwa CloudTrail data untuk `ChangeMessageVisibility`.

```
{
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
      },
      "attributes": {
        "creationDate": "2023-11-07T22:13:06Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:59:24Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "ChangeMessageVisibility",
  "awsRegion": "ap-southeast-4",
```

```

"sourceIPAddress": "10.0.118.80",
"userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
"requestParameters": {
  "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue",
  "receiptHandle": "AQEBy+2qnmQQVxcXrEwN7t6dXkjGAllr5DuSpGlvHx9s/vwbwp+RIr3dD6vRvlsU/
lteIulKHBs6DEIR7KL+J3mACfB+RRpRLWPlguiCdLKNKSVpdhkSBDDvkRHfycTHjuszGIebGdl+tYYjPrlz
+DSePmpty0EdhqtorW1xAc0Xf0GZbt0FtkbRFK3q151ETIHgthBCABoxu0CNvMElz9rYQ9m50Y30Z5Y0ZvQ/
coPHY1+9HhNV/A6Fs+/d6mVx9v6TomTh5L03wXqtjA8b0gkGftclQh/tJBAXqY/S8YG90KtY4NDP0SQBtYF/
vCCsCq9+5fiUfiYyvtdHSlwP9AyRotenCGrUKaRFiRhxDm1D6up0UaBs2d8wgHdKff/5mENTdeqrXQdZfwkFazW1a8ifWJm
+HzhfA9EJcdgWSS72WCMaerydsCxaX+E08B2ubL6oiafMYW4gK0GIRxYZ0+eeXKWy4TxkReW3j7k=",
  "visibilityTimeout": 1272
},
"responseElements": null,
"requestID": "6fbefbde-55d9-5640-98d1-a61a84457f14",
"eventID": "72275c61-bfc0-4606-934b-a6b7397aef20",
"readOnly": false,
"resources": [
  {
    "accountId": "123456789012",
    "type": "AWS::SQS::Queue",
    "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
  }
],
"eventType": "AwsApiCall",
"managementEvent": false,
"recipientAccountId": "123456789012",
"eventCategory": "Data",
"tlsDetails": {
  "tlsVersion": "TLSv1.2",
  "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
  "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
}
}

```

ChangeMessageVisibilityBatch

Contoh berikut menunjukkan peristiwa CloudTrail data untukChangeMessageVisibilityBatch.

```

{
  "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":
"AQEB2M5cVYg5gslhwME6537hdjcaPn0YPA5M0W460TTb0DzP1e631yPwm8qxd401hDj/
B4ntTMnsgBTa95t14tNx7Vn96jKJ5rIoZ7iI8TRmkT1caKodKIPs8w9yndZq50c2FPQxtyH+2L3UHf/
abV3szqVWX0LZR4PwX8zZkVWQGNCNnY2q21GCG586F8Qwvr0FYoXNwB8ymd1t77e1PDPknq1Io3JFuzkEsndkkETy4fV1Qc
15PHX17nXxaC+DURV1MPX0uSFACGmWqAoyk50HKwG0jLQgpySL/
TcnQXC1vFq8kNXGwyVzJsbwHp0HxI7oce69vaD6DaWFP75d3hx+PJeG9pauQCKzVP3skt3Hw/
zDC7YfKcALD3aCwMmeNDwT3w0BUG6XZdG51YhtFtTQYV7YuS3i/
Jh3HShGbtm07JK0EFiPkxv2+XNaAX3gFEpbng6zamTanfyMXCJIigIAEqiyWHQ=",
      "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. CloudWatch menganggap antrian aktif hingga enam jam jika berisi pesan apa pun atau jika ada tindakan yang mengaksesnya.

Ketika antrian 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

- 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](#)
- [Membuat CloudWatch alarm untuk metrik Amazon SQS](#)
- [CloudWatch Metrik yang tersedia untuk Amazon SQS](#)


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

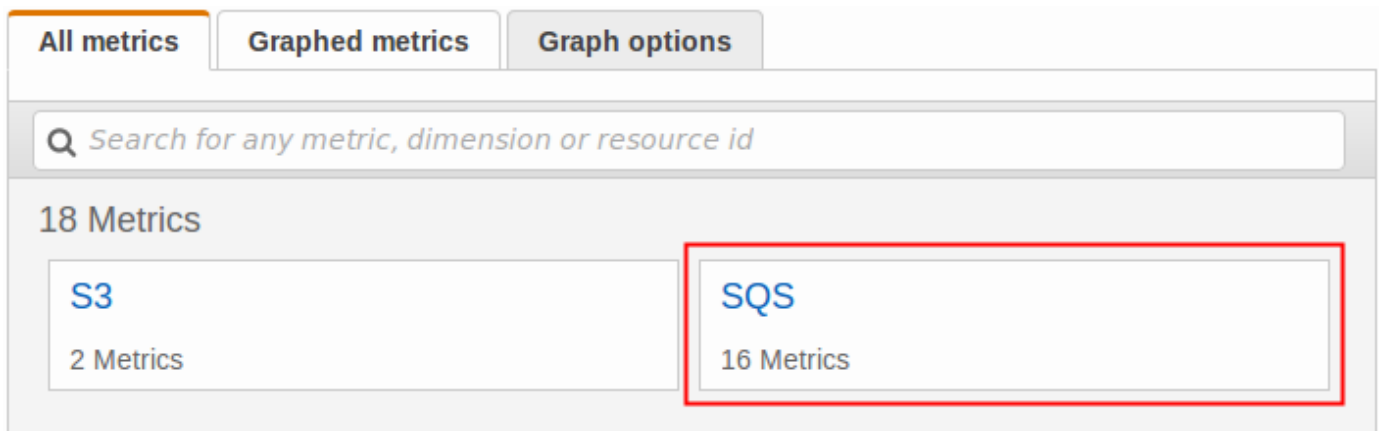
1. Masuk ke [konsol Amazon SQS](#).
2. Dalam daftar antrian, pilih (centang) kotak untuk antrian yang ingin Anda akses metrik. Anda dapat menampilkan metrik hingga 10 antrian.
3. Pilih tab Pemantauan.

Berbagai grafik ditampilkan di bagian metrik SQS.

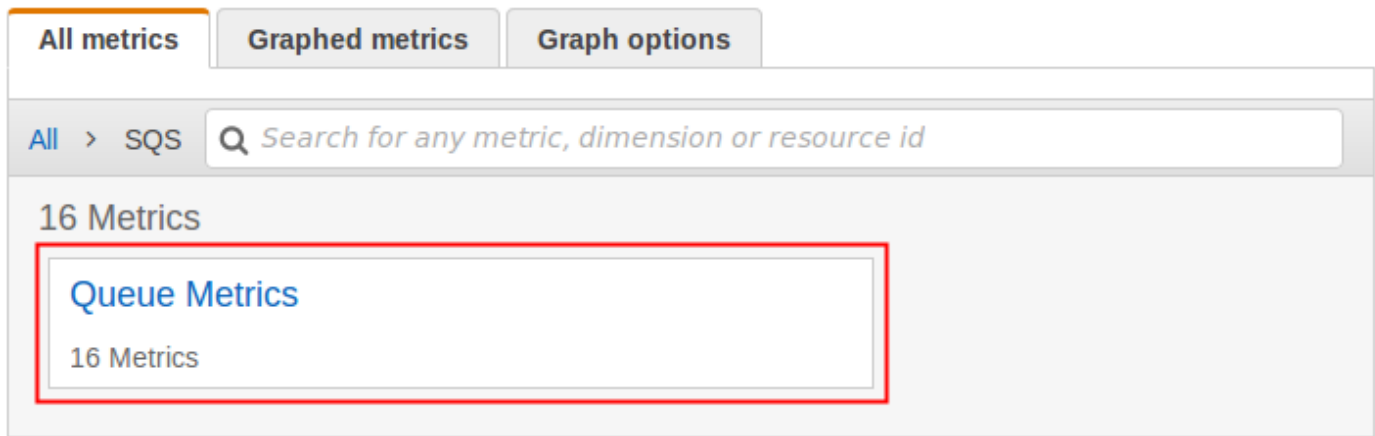
4. Untuk memahami apa yang diwakili oleh grafik tertentu, arahkan kursor ke  samping grafik yang diinginkan, atau lihat [CloudWatch Metrik yang tersedia untuk Amazon SQS](#).
5. Untuk mengubah rentang waktu untuk semua grafik secara bersamaan, untuk Rentang Waktu, pilih rentang waktu yang diinginkan (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.

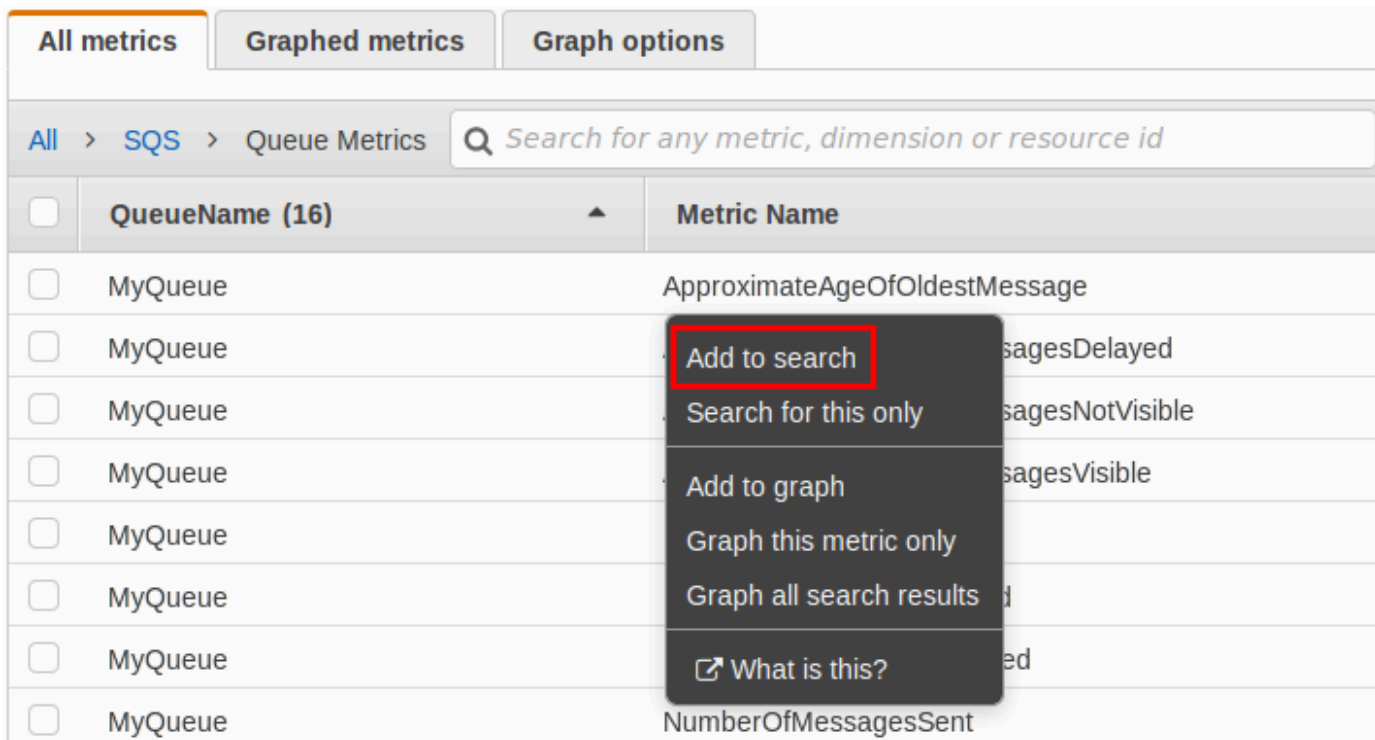


4. Pilih dimensi metrik Metrik Antrian.



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 yang akan 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 akan dikirimkan.

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
<code>ApproximateAgeOfOldestMessage</code>	Perkiraan usia pesan tertua yang tidak dihapus dalam antrian.

Metrik	Deskripsi
	<p data-bbox="938 247 1058 283"> Note</p> <ul data-bbox="987 331 1474 1812" style="list-style-type: none"><li data-bbox="987 331 1474 871">• Setelah pesan diterima tiga kali (atau lebih) dan tidak diproses, pesan dipindahkan ke bagian belakang antrian dan titik <code>ApproximateAgeOfOldestMessage</code> metrik pada pesan tertua kedua yang belum diterima lebih dari tiga kali. Tindakan ini terjadi bahkan jika antrian memiliki kebijakan <code>redrive</code>.<li data-bbox="987 898 1474 1249">• 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.<li data-bbox="987 1276 1474 1812">• Ketika antrian memiliki kebijakan <code>redrive</code>, pesan akan dipindahkan ke antrean huruf mati setelah jumlah maksimum penerima yang dikonfigurasi. Ketika pesan dipindahkan ke antrian huruf mati, <code>ApproximateAgeOfOldestMessage</code> metrik antrian huruf mati mewakili waktu ketika pesan dipindahkan ke antrian huruf


Metrik	Deskripsi
	<p>mati (bukan waktu asli pesan dikirim).</p> <ul style="list-style-type: none">• 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. <p>Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif.</p> <p>Unit: detik</p> <p>Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)</p>

Metrik	Deskripsi
<code>ApproximateNumberOfMessagesDelayed</code>	<p>Jumlah pesan dalam antrian yang tertunda dan tidak tersedia untuk dibaca segera. Ini dapat terjadi ketika antrian dikonfigurasi sebagai antrian penundaan atau ketika pesan telah dikirim dengan parameter penundaan.</p> <p>Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif.</p> <p>Unit: Hitungan</p> <p>Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)</p>
<code>ApproximateNumberOfMessagesNotVisible</code>	<p>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.</p> <p>Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif.</p> <p>Unit: Hitungan</p> <p>Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)</p>

Metrik	Deskripsi
<code>ApproximateNumberOfMessagesVisible</code>	<p>Jumlah pesan yang akan diproses.</p> <p>Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif.</p> <p>Unit: Hitungan</p> <p>Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)</p> <p>Tidak ada batasan jumlah pesan untuk proses, namun Anda dapat membuat backlog ini menjadi periode retensi.</p>
<code>NumberOfEmptyReceives</code> ¹	<p>Jumlah panggilan <code>ReceiveMessage</code> API yang tidak mengembalikan pesan.</p> <p>Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif.</p> <p>Unit: Hitungan</p> <p>Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)</p>

Metrik	Deskripsi
NumberOfMessagesDeleted ¹	<p>Jumlah pesan yang dihapus dari antrian.</p> <p>Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif.</p> <p>Unit: Hitungan</p> <p>Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)</p> <p>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:</p> <ul style="list-style-type: none">• 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
<p><code>NumberOfMessagesReceived</code> ¹</p>	<p>tetapi Anda memanggil <code>DeleteMessage</code> tindakan lagi menggunakan pegangan tanda terima yang sama, status keberhasilan akan dikembalikan, meningkatkan nilai <code>NumberOfMessagesDeleted</code> metrik.</p> <p>Jumlah pesan yang dikembalikan oleh panggilan ke <code>ReceiveMessage</code> tindakan.</p> <p>Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif.</p> <p>Unit: Hitungan</p> <p>Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)</p>
<p><code>NumberOfMessagesSent</code> ¹</p>	<p>Jumlah pesan yang ditambahkan ke antrian.</p> <p>Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif.</p> <p>Unit: Hitungan</p> <p>Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)</p>

Metrik	Deskripsi
SentMessageSize ¹	<p>Ukuran pesan yang ditambahkan ke antrian.</p> <p>Kriteria Pelaporan: Nilai non-negatif dilaporkan jika antrian aktif.</p> <p>Satuan: Byte</p> <p>Statistik yang Valid: Rata-rata, Minimum, Maksimum, Jumlah, Sampel Data (ditampilkan sebagai Jumlah Sampel di konsol Amazon SQS)</p> <div data-bbox="906 779 1507 1140" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px;"><p> Note</p><p>SentMessageSize tidak ditampilkan sebagai metrik yang tersedia di CloudWatch konsol sampai setidaknya satu pesan dikirim ke antrian yang sesuai.</p></div>

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 khusus, lihat [Layanan AWS di Scope oleh Program](#) Program Kepatuhan yang Anda minati. Untuk informasi umum, lihat [Program Kepatuhan 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 berdasarkan sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, serta hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- [Panduan Mulai Cepat Keamanan dan Kepatuhan](#) – Panduan deployment ini membahas pertimbangan arsitektur dan menyediakan langkah-langkah untuk melakukan deployment lingkungan dasar di AWS yang menjadi fokus keamanan dan kepatuhan.
- [Merancang Keamanan dan Kepatuhan HIPAA di Amazon Web Services](#) – Laporan resmi ini menjelaskan cara perusahaan dapat menggunakan AWS untuk membuat aplikasi yang memenuhi syarat HIPAA.

 Note

Tidak semua Layanan AWS memenuhi syarat HIPAA. Untuk informasi selengkapnya, lihat [Referensi Layanan yang Memenuhi Syarat HIPAA](#).

- [Sumber Daya Kepatuhan AWS](#) – Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- [Panduan Kepatuhan Pelanggan AWS](#) – Pahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan kontrol keamanan di banyak kerangka kerja (termasuk National Institute of Standards and Technology (NIST), Payment Card Industry Security Standards Council (PCI), dan International Organization for Standardization (ISO)).
- [Mengevaluasi Sumber Daya dengan Aturan](#) dalam Panduan Developer AWS Config – Layanan AWS Config menilai seberapa baik konfigurasi sumber daya Anda dalam mematuhi praktik-praktik internal, pedoman industri, dan regulasi internal.
- [AWS Security Hub](#) – Layanan AWS ini memberikan pandangan komprehensif tentang status keamanan Anda di dalam AWS. Security Hub menggunakan kontrol keamanan untuk mengevaluasi sumber daya AWS Anda dan memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat [Referensi kontrol Security Hub](#).
- [AWS Audit Manager](#) – Layanan AWS ini akan membantu Anda untuk terus-menerus mengaudit penggunaan AWS untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap regulasi dan standar industri.

Ketahanan di Amazon SQS

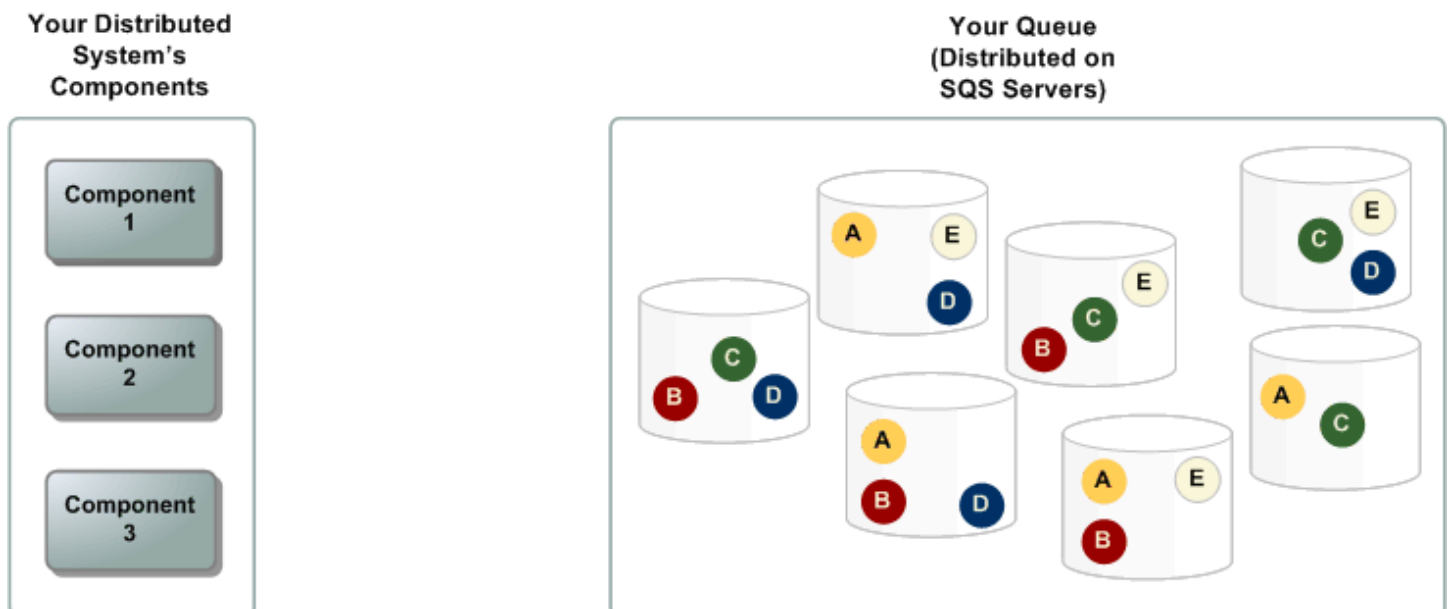
Infrastruktur global AWS dibangun di sekitar Wilayah AWS dan Availability Zone. AWS Wilayah menyediakan beberapa Availability Zone yang terpisah dan terisolasi secara fisik, yang terhubung dengan jaringan yang memiliki latensi rendah, throughput tinggi, dan sangat berlebihan. Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan fail over di antara zona tanpa gangguan. Availability Zone lebih tersedia, memiliki toleransi kesalahan, dan dapat diskalakan dibandingkan dengan satu atau beberapa infrastruktur pusat data tradisional. Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat [AWS Infrastruktur 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) secara berlebihan menyimpan pesan di beberapa server Amazon SQS.



Keamanan infrastruktur di Amazon SQS

Sebagai suatu layanan terkelola, Amazon SQS dilindungi oleh prosedur keamanan jaringan AWS global yang dijelaskan dalam laporan resmi [Amazon Web Services: Gambaran Umum Proses Keamanan](#).

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 access key ID dan secret access key yang terhubung dengan prinsipal IAM. Atau, Anda dapat menggunakan [AWS Security Token Service](#) (AWS STS) untuk menghasilkan kredensial keamanan sementara untuk permintaan penandatanganan.

Anda dapat menghubungi 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 Amazon VPC endpoint tertentu atau VPC tertentu. Secara efektif, dapat mengisolasi akses jaringan ke antrean Amazon SQS yang diberikan hanya dari 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 ditinjau dalam konteks kebijakan keamanan Anda.

Note

Panduan implementasi khusus yang disediakan untuk kasus penggunaan dan implementasi umum. Sebaiknya Anda melihat praktik terbaik ini dalam konteks kasus penggunaan, arsitektur, dan model ancaman tertentu Anda.

Praktik terbaik pencegahan

Berikut adalah praktik terbaik keamanan pencegahan untuk Amazon SQS.

Topik

- [Memastikan bahwa antrian tidak dapat diakses secara publik](#)
- [Menerapkan akses hak istimewa yang paling rendah](#)
- [Gunakan peran IAM untuk aplikasi dan AWS layanan yang memerlukan akses Amazon SQS](#)
- [Menerapkan enkripsi sisi server](#)
- [Menegakkan enkripsi data saat transit](#)
- [Pertimbangkan titik akhir VPC untuk mengakses Amazon SQS](#)

Memastikan bahwa antrian tidak dapat diakses secara publik

Kecuali jika Anda meminta siapa pun secara eksplisit di internet untuk dapat membaca atau menulis ke antrian Amazon SQS, pastikan bahwa antrian Anda tidak dapat diakses publik (dapat diakses oleh semua orang di dunia atau oleh AWS pengguna yang diautentikasi).

- 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 apa izin tersebut, dan tindakan API tertentu yang ingin Anda izinkan untuk antrian ini. Menerapkan akses hak istimewa paling rendah 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.
- Produsen - 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

Agar aplikasi atau AWS layanan seperti Amazon EC2 untuk mengakses antrian Amazon SQS, mereka harus menggunakan AWS kredensi yang valid dalam permintaan AWS API. Karena kredensial ini tidak diputar secara otomatis, sebaiknya Anda tidak menyimpan kredensial AWS secara langsung di aplikasi atau instans EC2.

Anda harus menggunakan IAM role untuk mengelola kredensial sementara untuk aplikasi atau layanan yang perlu mengakses Amazon SQS. Saat Anda menggunakan peran, Anda tidak perlu mendistribusikan kredensial jangka panjang (seperti nama pengguna dan kata sandi atau access key) ke instans EC2 atau AWS layanan seperti AWS Lambda. Sebaliknya, peran menyediakan izin sementara yang dapat digunakan aplikasi saat mereka melakukan panggilan ke sumber daya AWS lainnya.

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 di tingkat pesan saat menyimpannya, dan mendekripsi pesan untuk Anda saat mengaksesnya. SSE menggunakan kunci yang dikelola di AWS Key Management Service. Selama Anda mengautentikasi permintaan Anda dan memiliki izin akses, tidak ada perbedaan dalam mengakses antrian terenkripsi atau tidak terenkripsi.

Untuk informasi selengkapnya, lihat [Enkripsi diam](#) dan [Manajemen kunci](#).

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 koneksi terenkripsi

melalui HTTPS (TLS) menggunakan [aws:SecureTransport](#) kondisi dalam kebijakan antrian untuk memaksa permintaan menggunakan SSL.

Pertimbangkan titik akhir VPC untuk mengakses Amazon SQS

Jika Anda memiliki antrian yang harus berinteraksi dengan Anda tetapi harus benar-benar tidak terekspos 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 Amazon VPC tertentu atau dari VPC tertentu.

Titik akhir VPC Amazon SQS menyediakan dua cara untuk mengontrol akses ke pesan Anda:

- Anda dapat mengontrol permintaan, pengguna, atau grup yang diizinkan melalui VPC endpoint tertentu.
- Anda dapat mengendalikan VPC atau VPC endpoint mana yang memiliki akses ke antrian Anda menggunakan kebijakan antrian.

Selengkapnya, lihat [Titik akhir Amazon Virtual Private Cloud untuk Amazon SQS](#) dan [Membuat kebijakan titik akhir Amazon VPC untuk Amazon SQS](#).

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 [AWSSDK](#), 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 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.

- AWSProtokol 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](#)

AWSProtokol JSON tersedia pada versi [AWSSDK](#) yang ditentukan. Untuk meninjau versi SDK dan tanggal rilis di seluruh varian bahasa, lihat [AWSmatriks dukungan versi SDK dan Alat](#) di Panduan Referensi AWSSDK dan Alat

Topik

- [Membuat permintaan API kueri menggunakan protokol AWS JSON](#)
- [Membuat permintaan Query API dengan protokol AWS kueri](#)
- [Mengautentikasi permintaan](#)
- [Tindakan batch Amazon SQS](#)

Membuat permintaan API kueri menggunakan protokol AWS JSON

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

Note

AWSProtokol JSON didukung untuk sebagian besar varian bahasa. Untuk daftar lengkap varian bahasa yang didukung, lihat [Bahasa apa yang didukung untuk protokol AWS JSON yang digunakan di Amazon SQS API?](#)

Topik

- [Membangun titik akhir](#)
- [Membuat permintaan POST](#)
- [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 `MyQueue` dan 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 keAmazonSQS.<operationName>, dan header HTTP dengan Content-Type set keapplication/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](#) dan [FAQ protokol Amazon SQS AWS JSON](#).

Topik

- [Struktur respons JSON yang sukses](#)

- [Struktur respons kesalahan JSON](#)

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++

Bahasa	Repositori klien SDK	Versi klien SDK yang diperlukan	Panduan
Golang 1.x	aws/ aws-sdk-go	v1.47.7	AWS SDK for Go.
Golang 2.x	aws/ 2 aws-sdk-go-v	v1.28.0	AWSSDK for Go V2
Java 1.x	aws/ aws-sdk-java	1.12.585	AWS SDK for Java.
Java 2.x	aws/ 2 aws-sdk-java-v	2.21.19	AWS SDK for Java.
JavaScript v2.x	aws/ aws-sdk-js	v2.1492.0	JavaScript pada AWS
JavaScript v3.x	aws/ 3 aws-sdk-js-v	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	AWSSDK for PHP
Python-Boto3	boto/boto3	1.28.82	AWSSDK untuk Python (Boto3)
Python-botocore	boto/botocore	1.31.82	AWSSDK untuk Python (Boto3)
awscli	AWSCLI	1.29.82	AWSAntarmuka Baris Perintah
Ruby	aws/ aws-sdk-ruby	1.67.0	AWSSDK 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 [AWSwilayah](#) 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

AWSProtokol 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. [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 [AWSJSON 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](#)

Membuat permintaan Query API dengan protokol AWS kueri

Di bagian ini Anda mempelajari cara membuat titik akhir Amazon SQS, membuat permintaan GET dan POST, dan menafsirkan respons.

Topik

- [Membangun titik akhir](#)
- [Membuat permintaan GET](#)
- [Membuat permintaan POST](#)
- [Menafsirkan respons API XML-Azon SQS](#)

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 Amazon Simple Queue Service](#)

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 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 bersifat case sensitive.

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

- Endpoint - Sumber daya yang digunakan permintaan ([nama antrian dan URL](#)), misalnya:
`https://sqs.us-east-2.amazonaws.com/123456789012/MyQueue`
- Tindakan - [Tindakan](#) yang ingin Anda lakukan pada titik akhir. Tanda tanya (?) memisahkan titik akhir dari aksi, 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 mengirimkan 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 bersifat case sensitive.

Karena permintaan GET adalah URL, Anda harus URL-encode semua nilai parameter. Karena spasi tidak diizinkan dalam URL, setiap ruang 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 dalam isi permintaan HTTP.

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

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

Header diikuti oleh permintaan [form-urlencoded](#) GET yang mengirimkan 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 respons API XML-Amazon SQS

Menanggapi permintaan tindakan, Amazon SQS mengembalikan struktur data XML yang berisi hasil permintaan. Untuk informasi selengkapnya, lihat tindakan individual dalam Referensi [API Layanan Antrian Sederhana Amazon](#).

Topik

- [Struktur respon XML-sukses](#)
- [Struktur respon kesalahan XML-](#)

Struktur respon XML-sukses

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

Elemen ini berisi elemen turunan berikut:

- **ActionNameResult**- Berisi elemen tindakan-spesifik. 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 ini berisi elemen anak berikut:

- **Type**- Menentukan apakah kesalahan adalah kesalahan produsen atau konsumen.
- **Code**- Menentukan jenis kesalahan.

- **Message**- Menentukan kondisi kesalahan dalam format yang dapat dibaca.
- **Detail**- (Opsional) Menentukan rincian tambahan tentang kesalahan.

RequestIdElemen 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

Otentikasi adalah proses mengidentifikasi dan memverifikasi pihak yang mengirimkan permintaan. Selama tahap pertama otentikasi, AWS memverifikasi identitas produsen dan apakah produsen [terdaftar untuk digunakan AWS](#) (untuk informasi lebih lanjut, lihat [Langkah 1: Buat pengguna Akun AWS dan IAM](#)). Berikutnya, AWS mematuhi 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 tersebut.
4. Salah satu dari yang berikut terjadi:
 - Jika otentikasi berhasil, konsumen memproses permintaan.
 - Jika otentikasi gagal, konsumen menolak permintaan dan mengembalikan kesalahan.

Topik

- [Proses otentikasi dasar dengan HMAC-SHA](#)

- [Bagian 1: Permintaan dari pengguna](#)
- [Bagian 2: Tanggapan dari AWS](#)

Proses otentikasi dasar dengan HMAC-SHA

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

- AWS Access Key ID yang mengidentifikasi Akun AWS, yang 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 lebih lanjut, 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. Turunkan Signature Version 4 Signing Key. Untuk informasi selengkapnya, lihat [Menurunkan Kunci Penandatanganan dengan Java](#).

Note

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

2. Base64 mengodekan 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 berupadateTime objek, dengan [tanggal lengkap, termasuk jam, menit, dan detik](#). Sebagai contoh:2007-01-31T23:59:59Z Meskipun ini tidak diperlukan, kami sarankan untuk menyediakan objek menggunakan zona waktu Coordinated Universal Time (Greenwich Mean Time).

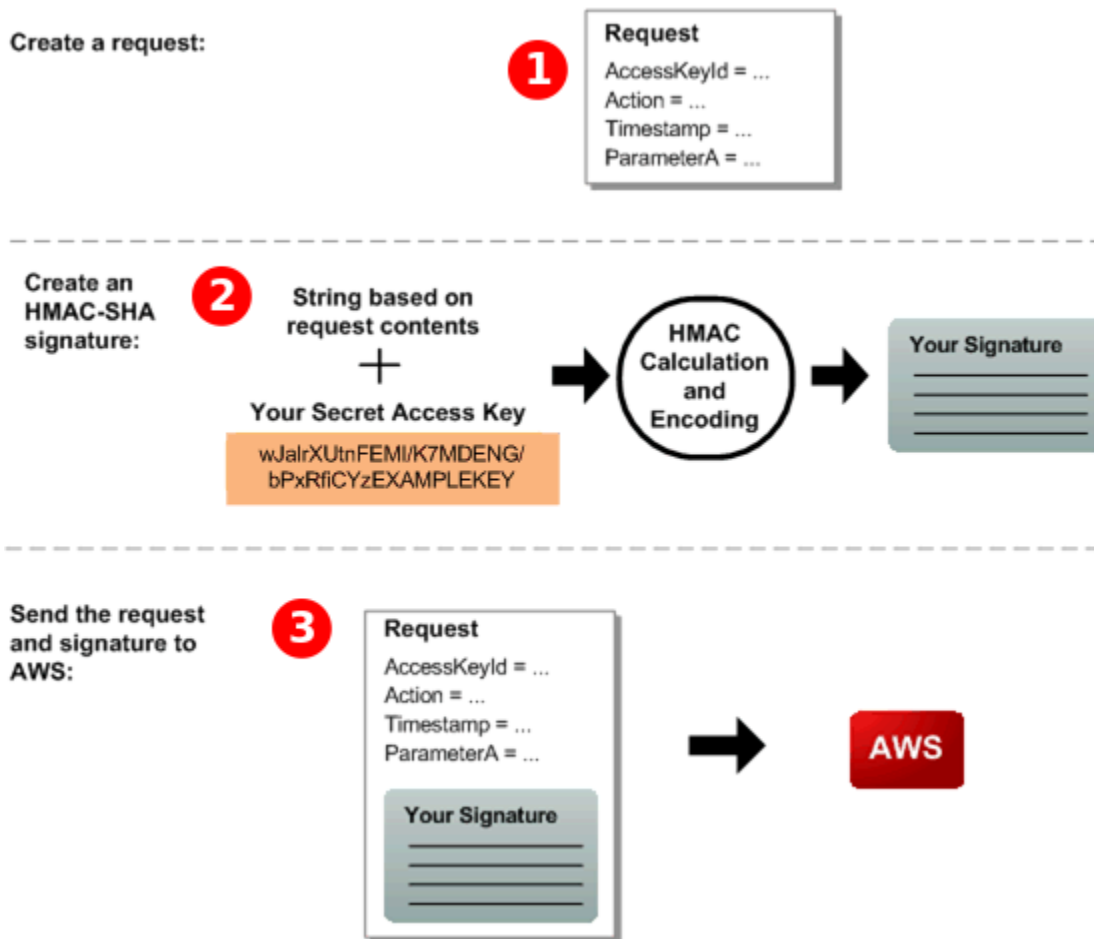
Note

Verifikasi bahwa waktu server Anda diatur dengan benar. Jika Anda menentukan stempel waktu (bukan kedaluwarsa), permintaan secara otomatis akan berakhir 15 menit setelah waktu yang ditentukan (AWS tidak memproses permintaan dengan stempel waktu lebih dari 15 menit lebih awal dari waktu saat ini diAWS server).

Jika Anda menggunakan .NET, Anda tidak boleh mengirim stempel waktu yang terlalu spesifik (karena interpretasi yang berbeda tentang bagaimana ketepatan waktu ekstra harus dijatuhkan). Dalam hal ini, Anda harus secara manual membangundateTime objek dengan presisi tidak lebih dari satu milidetik.

Bagian 1: Permintaan dari pengguna

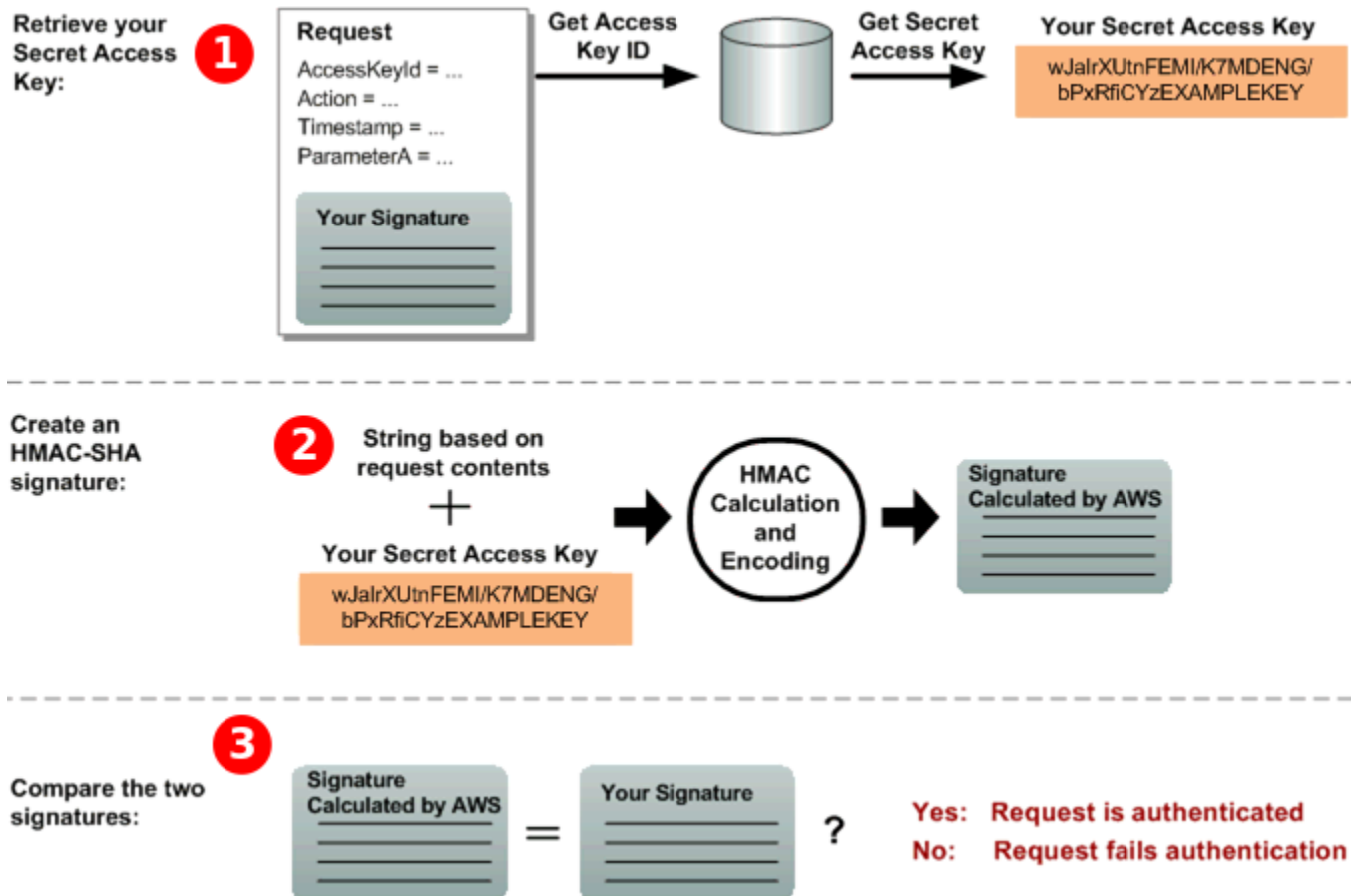
Berikut ini adalah proses yang harus Anda ikuti untuk mengautentikasiAWS permintaan menggunakan tanda tangan permintaan HMAC-SHA.



1. Bangun permintaan keAWS.
2. Hitung tanda tangan kode otentikasi pesan hash keyed-hash (HMAC-SHA) menggunakan Kunci Akses Rahasia Anda.
3. Sertakan tanda tangan dan ID Kunci Akses Anda dalam permintaan, lalu kirim permintaan keAWS.

Bagian 2: Tanggapan dariAWS

AWSmemulai proses berikut sebagai tanggapan.



1. AWS menggunakan Access Key untuk mencari Secret Access Key.
2. AWS menghasilkan tanda tangan dari data permintaan dan Secret Access Key, menggunakan algoritma yang sama yang Anda gunakan untuk menghitung tanda tangan yang Anda kirim dalam permintaan.
3. Salah satu dari yang berikut terjadi:
 - Jika tanda tangan yang AWS dihasilkan cocok dengan yang Anda kirim dalam permintaan, AWS anggap permintaan itu otentik.
 - Jika perbandingan gagal, maka permintaan dibuang, dan AWS menghasilkan 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 `SendMessageBatch` panggilan tidak dapat melebihi 262.144 byte (256 KB).

Anda tidak dapat mengatur izin untuk `SendMessageBatch`, `DeleteMessageBatch`, atau `ChangeMessageVisibilityBatch` secara eksplisit. Mengatur izin untuk `SendMessage`, `DeleteMessage`, atau `ChangeMessageVisibility` menetapkan izin untuk versi batch yang sesuai dari tindakan.

Konsol Amazon SQS tidak mendukung tindakan batch.

Topik

- [Mengaktifkan buffering sisi klien dan batching permintaan](#)
- [Meningkatkan throughput menggunakan penskalaan horizontal dan tindakan batching](#)

Mengaktifkan buffering sisi klien dan batching permintaan

Klaster [AWS SDK for Java](#) termasuk `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. Batched permintaan dan dukungan untuk [polling](#) juga dapat membantu meningkatkan throughput. Untuk informasi selengkapnya, lihat [Meningkatkan throughput menggunakan penskalaan horizontal dan tindakan batching](#).

Karena `AmazonSQSBufferedAsyncClient` mengimplementasikan antarmuka yang sama seperti `AmazonSQSAsyncClient`, migrasi

dari `AmazonSQSAsyncClient` kepada `AmazonSQSBufferedAsyncClient` biasanya hanya membutuhkan sedikit perubahan pada kode yang ada.

Note

Amazon SQS Buffered Asynchronous Client saat ini tidak mendukung antrean FIFO.

Topik

- [Menggunakan AmazonSQSBufferedAsyncClient](#)
- [Mengonfigurasi AmazonSQSBufferedAsyncClient](#)

Menggunakan AmazonSQSBufferedAsyncClient

Sebelum memulai, selesaikan langkah-langkah di [Menyiapkan Amazon SQS](#).

Important

Klaster AWS SDK for Java 2.x belum kompatibel dengan `AmazonSQSBufferedAsyncClient`.

Anda dapat membuat `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 `AmazonSQSBufferedAsyncClient`, Anda dapat menggunakannya untuk mengirim beberapa permintaan ke Amazon SQS (seperti yang Anda bisa `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);
```

Mengonfigurasi AmazonSQSBufferedAsyncClient

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

1. Buatlah sebuah instans 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

Parameter	Nilai default	Deskripsi
longPoll	true	Saat longPoll diatur ke true, AmazonSQS


Parameter	Nilai default	Deskripsi
		<code>BufferedAsyncClient</code> mencoba untuk menggunakan polling panjang ketika mengkonsumsi pesan.
<code>longPollWaitTimeoutSeconds</code>	20 dtk	<p>Jumlah maksimum waktu (dalam satuan detik)<code>ReceiveMessage</code> blok panggilan off pada server, menunggu pesan muncul dalam antrian sebelum kembali dengan hasil menerima kosong.</p> <div data-bbox="1068 877 1507 1192"><p> Note</p><p>Ketika polling panjang dinonaktifkan, pengaturan ini tidak berpengaruh.</p></div>

Parameter	Nilai default	Deskripsi
maxBatchOpenMs	200	<p>Jumlah maksimum waktu (dalam milidetik) bahwa panggilan keluar menunggu panggilan lain yang dengannya ia batch pesan dari jenis yang sama.</p> <p>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).</p> <p>Saat Anda mengatur parameter ini ke 0, permintaan yang dikirimkan tidak menunggu permintaan lain, secara efektif menonaktifkan batching.</p>

Parameter	Nilai default	Deskripsi
<code>maxBatchSize</code>	10 permintaan per	<p>Jumlah pesan maksimum yang dikumpulkan bersama dalam satu permintaan. Semakin tinggi pengaturan, semakin sedikit batch yang diperlukan untuk melakukan jumlah permintaan yang sama.</p> <div data-bbox="1068 621 1507 982"><p> Note</p><p>10 permintaan per batch adalah nilai maksimum yang diizinkan untuk Amazon SQS.</p></div>
<code>maxBatchSizeBytes</code>	256 KB	<p>Ukuran maksimum kumpulan pesan, dalam byte, yang coba dikirim klien ke Amazon SQS.</p> <div data-bbox="1068 1289 1507 1604"><p> Note</p><p>256 KB adalah nilai maksimum yang diizinkan untuk Amazon SQS.</p></div>

Parameter	Nilai default	Deskripsi
<code>maxDoneReceiveBatches</code>	10	<p>Jumlah maksimum menerima AmazonSQS <code>BufferedAsyncClient</code> prefetches dan toko sisi klien.</p> <p>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).</p> <div data-bbox="1068 1129 1507 1541" style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px;"><p> Note</p><p>0 menunjukkan bahwa semua prefetching pesan dinonaktifkan dan pesan hanya dikonsumsi sesuai permintaan.</p></div>

Parameter	Nilai default	Deskripsi
<code>maxInflightOutboundBatches</code>	5	<p>Jumlah maksimum batch keluar aktif yang dapat diproses pada waktu yang sama.</p> <p>Semakin tinggi pengaturannya, semakin cepat batch keluar dapat dikirim (tunduk pada kuota seperti CPU atau bandwidth) dan semakin banyak thread yang dikonsumsi oleh <code>AmazonSQS.BufferedAsyncClient</code>.</p>

Parameter	Nilai default	Deskripsi
maxInflightReceive Batches	10	<p>Jumlah maksimum menerima aktif yang dapat diproses pada waktu yang sama.</p> <p>Semakin tinggi pengaturan, semakin banyak pesan yang dapat diterima (tunduk pada kuota seperti CPU atau bandwidth), dan semakin banyak utas yang dikonsumsi <code>AmazonSQSBufferedAsyncClient</code>.</p> <div data-bbox="1068 844 1507 1255"><p> Note</p><p>0 menunjukkan bahwa semua prefetching pesan dinonaktifkan dan pesan hanya dikonsumsi sesuai permintaan.</p></div>

Parameter	Nilai default	Deskripsi
<code>visibilityTimeoutSeconds</code>	-1	<p>Bila parameter ini disetel ke nilai positif dan bukan nol, batas waktu visibilitas yang ditetapkan di sini akan menggantikan batas waktu visibilitas yang ditetapkan pada antrian dari mana pesan digunakan.</p> <div data-bbox="1068 667 1510 1129"><p>Note</p><p>-1 menunjukkan bahwa pengaturan default dipilih untuk antrian. Anda tidak dapat mengatur batas waktu visibilitas 0.</p></div>

Meningkatkan throughput menggunakan penskalaan horizontal dan tindakan batching

Antrian Amazon SQS dapat memberikan throughput yang sangat tinggi. Untuk informasi tentang kuota throughput, lihat [Kuota yang terkait dengan pesan](#).

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

Topik

- [Penskalaan Horizontal](#)
- [Pembuatan batch](#)

- [Bekerja contoh Java untuk single-operasi dan batch permintaan](#)

Penskalaan Horizontal

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

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

- Tingkatkan jumlah utas per klien
- Tambahkan lebih banyak klien
- Tingkatkan jumlah utas per klien dan tambahkan lebih banyak klien

Ketika Anda menambahkan lebih banyak klien, Anda mencapai keuntungan dasarnya linear 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 pesan bersamaan dan konsumen yang mengirim permintaan dan menerima tanggapan. Misalnya, secara default, instans AWS SDK for Java [AmazonSQSClient](#) kelas mempertahankan paling banyak 50 koneksi ke Amazon SQS. Untuk membuat produsen bersamaan tambahan dan konsumen, Anda harus menyesuaikan jumlah maksimum produsen dan konsumen benang yang diijinkan pada [AmazonSQSClientBuilder](#) objek, misalnya:

```
final AmazonSQS sqsClient = AmazonSQSClientBuilder.standard()
    .withClientConfiguration(new ClientConfiguration()
        .withMaxConnections(producerCount + consumerCount))
    .build();
```


Untuk [AmazonSQSAsyncClient](#), Anda juga harus memastikan bahwa cukup utas tersedia. Contoh ini hanya bekerja untuk Java v 1.x.

Pembuatan batch

Pembuatan melakukan lebih banyak pekerjaan selama setiap perjalanan pulang pergi ke layanan (misalnya, ketika Anda mengirim beberapa pesan dengan satu `SendMessageBatch` request). Tindakan batch Amazon SQS adalah [SendMessageBatch](#), [DeleteMessageBatch](#), dan [ChangeMessageVisibilityBatch](#). Untuk memanfaatkan batching tanpa mengubah produsen atau konsumen Anda, Anda dapat menggunakan [Klien Asinkron Buffer Amazon SQS](#).

Note

Karena [ReceiveMessage](#) dapat memproses 10 pesan sekaligus, tidak ada `ReceiveMessageBatch` tindakan.

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

Anda dapat menggabungkan batching dengan penskalaan horizontal untuk memberikan 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 tersebut, pengelompokan dapat mengurangi biaya Anda secara substansif.

Batching dapat memperkenalkan beberapa kompleksitas untuk aplikasi Anda (misalnya, aplikasi Anda harus mengumpulkan pesan sebelum mengirimnya, atau kadang-kadang harus menunggu lebih lama untuk respons). Namun, batching dapat tetap efektif dalam 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 kontrol.

⚠ 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.

Bekerja contoh Java untuk single-operasi dan batch permintaan

Prasyarat

Tambahkan `aws-java-sdk-sqs.jar`, `aws-java-sdk-ec2.jar`, dan `commons-logging.jar` paket ke jalur kelas 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 produser-konsumen sederhana. Utas utama memunculkan sejumlah utas 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 2010-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.
*
* Licensed under the Apache License, Version 2.0 (the "License").
* You may not use this file except in compliance with the License.
* A copy of the License is located at
*
* https://aws.amazon.com/apache2.0
*
* or in the "license" file accompanying this file. This file is distributed
* on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
* express or implied. See the License for the specific language governing
* permissions and limitations under the License.
*
*/
```

```
import com.amazonaws.AmazonClientException;
import com.amazonaws.ClientConfiguration;
import com.amazonaws.services.sqs.AmazonSQS;
import com.amazonaws.services.sqs.AmazonSQSClientBuilder;
import com.amazonaws.services.sqs.model.*;
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;

import java.math.BigInteger;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import java.util.Scanner;
import java.util.concurrent.TimeUnit;
import java.util.concurrent.atomic.AtomicBoolean;
import java.util.concurrent.atomic.AtomicInteger;

/**
 * Start a specified number of producer and consumer threads, and produce-consume
 * for the least of the specified duration and 1 hour. Some messages can be left
 * in the queue because producers and consumers might not be in exact balance.
 */
public class SimpleProducerConsumer {

    // The maximum runtime of the program.
    private final static int MAX_RUNTIME_MINUTES = 60;
    private final static Log log = LogFactory.getLog(SimpleProducerConsumer.class);

    public static void main(String[] args) throws InterruptedException {
```

```
final Scanner input = new Scanner(System.in);

System.out.print("Enter the queue name: ");
final String queueName = input.nextLine();

System.out.print("Enter the number of producers: ");
final int producerCount = input.nextInt();

System.out.print("Enter the number of consumers: ");
final int consumerCount = input.nextInt();

System.out.print("Enter the number of messages per batch: ");
final int batchSize = input.nextInt();

System.out.print("Enter the message size in bytes: ");
final int messageSizeByte = input.nextInt();

System.out.print("Enter the run time in minutes:");
final int runTimeMinutes = input.nextInt();

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

final AmazonSQS sqsClient = AmazonSQSClientBuilder.standard()
    .withClientConfiguration(clientConfiguration)
    .build();

final String queueUrl = sqsClient
    .getQueueUrl(new GetQueueUrlRequest(queueName)).getQueueUrl();

// The flag used to stop producer, consumer, and monitor threads.
final AtomicBoolean stop = new AtomicBoolean(false);

// Start the producers.
final AtomicInteger producedCount = new AtomicInteger();
final Thread[] producers = new Thread[producerCount];
for (int i = 0; i < producerCount; i++) {
    if (batchSize == 1) {
```

```
        producers[i] = new Producer(sqsClient, queueUrl, messageSizeByte,
                                   producedCount, stop);
    } else {
        producers[i] = new BatchProducer(sqsClient, queueUrl, batchSize,
                                         messageSizeByte, producedCount,
                                         stop);
    }
    producers[i].start();
}

// Start the consumers.
final AtomicInteger consumedCount = new AtomicInteger();
final Thread[] consumers = new Thread[consumerCount];
for (int i = 0; i < consumerCount; i++) {
    if (batchSize == 1) {
        consumers[i] = new Consumer(sqsClient, queueUrl, consumedCount,
                                    stop);
    } else {
        consumers[i] = new BatchConsumer(sqsClient, queueUrl, batchSize,
                                         consumedCount, stop);
    }
    consumers[i].start();
}

// Start the monitor thread.
final Thread monitor = new Monitor(producedCount, consumedCount, stop);
monitor.start();

// Wait for the specified amount of time then stop.
Thread.sleep(TimeUnit.MINUTES.toMillis(Math.min(runtimeMinutes,
                                                MAX_RUNTIME_MINUTES)));
stop.set(true);

// Join all threads.
for (int i = 0; i < producerCount; i++) {
    producers[i].join();
}

for (int i = 0; i < consumerCount; i++) {
    consumers[i].join();
}

monitor.interrupt();
monitor.join();
```

```
}

private static String makeRandomString(int sizeByte) {
    final byte[] bs = new byte[(int) Math.ceil(sizeByte * 5 / 8)];
    new Random().nextBytes(bs);
    bs[0] = (byte) ((bs[0] | 64) & 127);
    return new BigInteger(bs).toString(32);
}

/**
 * The producer thread uses {@code SendMessage}
 * to send messages until it is stopped.
 */
private static class Producer extends Thread {
    final AmazonSQS sqsClient;
    final String queueUrl;
    final AtomicInteger producedCount;
    final AtomicBoolean stop;
    final String theMessage;

    Producer(AmazonSQS sqsQueueBuffer, String queueUrl, int messageSizeByte,
            AtomicInteger producedCount, AtomicBoolean stop) {
        this.sqsClient = sqsQueueBuffer;
        this.queueUrl = queueUrl;
        this.producedCount = producedCount;
        this.stop = stop;
        this.theMessage = makeRandomString(messageSizeByte);
    }

    /**
     * The producedCount object tracks the number of messages produced by
     * all producer threads. If there is an error, the program exits the
     * run() method.
     */
    public void run() {
        try {
            while (!stop.get()) {
                sqsClient.sendMessage(new SendMessageRequest(queueUrl,
                    theMessage));
                producedCount.incrementAndGet();
            }
        } catch (AmazonClientException e) {
            /**
             * By default, AmazonSQSClient retries calls 3 times before

```

```
        * failing. If this unlikely condition occurs, stop.
        */
        log.error("Producer: " + e.getMessage());
        System.exit(1);
    }
}

/**
 * The producer thread uses {@code SendMessageBatch}
 * to send messages until it is stopped.
 */
private static class BatchProducer extends Thread {
    final AmazonSQS sqsClient;
    final String queueUrl;
    final int batchSize;
    final AtomicInteger producedCount;
    final AtomicBoolean stop;
    final String theMessage;

    BatchProducer(AmazonSQS sqsQueueBuffer, String queueUrl, int batchSize,
        int messageSizeByte, AtomicInteger producedCount,
        AtomicBoolean stop) {
        this.sqsClient = sqsQueueBuffer;
        this.queueUrl = queueUrl;
        this.batchSize = batchSize;
        this.producedCount = producedCount;
        this.stop = stop;
        this.theMessage = makeRandomString(messageSizeByte);
    }

    public void run() {
        try {
            while (!stop.get()) {
                final SendMessageBatchRequest batchRequest =
                    new SendMessageBatchRequest().withQueueUrl(queueUrl);

                final List<SendMessageBatchRequestEntry> entries =
                    new ArrayList<SendMessageBatchRequestEntry>();
                for (int i = 0; i < batchSize; i++)
                    entries.add(new SendMessageBatchRequestEntry()
                        .withId(Integer.toString(i))
                        .withMessageBody(theMessage));
                batchRequest.setEntries(entries);
            }
        }
    }
}
```

```
        final SendMessageBatchResult batchResult =
            sqsClient.sendMessageBatch(batchRequest);
        producedCount.addAndGet(batchResult.getSuccessful().size());

        /*
         * Because SendMessageBatch can return successfully, but
         * individual batch items fail, retry the failed batch items.
         */
        if (!batchResult.getFailed().isEmpty()) {
            log.warn("Producer: retrying sending "
                + batchResult.getFailed().size() + " messages");
            for (int i = 0, n = batchResult.getFailed().size();
                i < n; i++) {
                sqsClient.sendMessage(new
                    SendMessageRequest(queueUrl, theMessage));
                producedCount.incrementAndGet();
            }
        }
    } catch (AmazonClientException e) {
        /*
         * By default, AmazonSQSClient retries calls 3 times before
         * failing. If this unlikely condition occurs, stop.
         */
        log.error("BatchProducer: " + e.getMessage());
        System.exit(1);
    }
}

/**
 * The consumer thread uses {@code ReceiveMessage} and {@code DeleteMessage}
 * to consume messages until it is stopped.
 */
private static class Consumer extends Thread {
    final AmazonSQS sqsClient;
    final String queueUrl;
    final AtomicInteger consumedCount;
    final AtomicBoolean stop;

    Consumer(AmazonSQS sqsClient, String queueUrl, AtomicInteger consumedCount,
        AtomicBoolean stop) {
        this.sqsClient = sqsClient;
    }
}
```



```
        this.queueUrl = queueUrl;
        this.consumedCount = consumedCount;
        this.stop = stop;
    }

    /**
     * Each consumer thread receives and deletes messages until the main
     * thread stops the consumer thread. The consumedCount object tracks the
     * number of messages that are consumed by all consumer threads, and the
     * count is logged periodically.
     */
    public void run() {
        try {
            while (!stop.get()) {
                try {
                    final ReceiveMessageResult result = sqsClient
                        .receiveMessage(new
                            ReceiveMessageRequest(queueUrl));

                    if (!result.getMessages().isEmpty()) {
                        final Message m = result.getMessages().get(0);
                        sqsClient.deleteMessage(new
                            DeleteMessageRequest(queueUrl,
                                m.getReceiptHandle()));
                        consumedCount.incrementAndGet();
                    }
                } catch (AmazonClientException e) {
                    log.error(e.getMessage());
                }
            }
        } catch (AmazonClientException e) {
            /**
             * By default, AmazonSQSClient retries calls 3 times before
             * failing. If this unlikely condition occurs, stop.
             */
            log.error("Consumer: " + e.getMessage());
            System.exit(1);
        }
    }
}

/**
 * The consumer thread uses {@code ReceiveMessage} and {@code
 * DeleteMessageBatch} to consume messages until it is stopped.
 */
```

```
*/
private static class BatchConsumer extends Thread {
    final AmazonSQS sqsClient;
    final String queueUrl;
    final int batchSize;
    final AtomicInteger consumedCount;
    final AtomicBoolean stop;

    BatchConsumer(AmazonSQS sqsClient, String queueUrl, int batchSize,
        AtomicInteger consumedCount, AtomicBoolean stop) {
        this.sqsClient = sqsClient;
        this.queueUrl = queueUrl;
        this.batchSize = batchSize;
        this.consumedCount = consumedCount;
        this.stop = stop;
    }

    public void run() {
        try {
            while (!stop.get()) {
                final ReceiveMessageResult result = sqsClient
                    .receiveMessage(new ReceiveMessageRequest(queueUrl)
                        .withMaxNumberOfMessages(batchSize));

                if (!result.getMessages().isEmpty()) {
                    final List<Message> messages = result.getMessages();
                    final DeleteMessageBatchRequest batchRequest =
                        new DeleteMessageBatchRequest()
                            .withQueueUrl(queueUrl);

                    final List<DeleteMessageBatchRequestEntry> entries =
                        new ArrayList<DeleteMessageBatchRequestEntry>();
                    for (int i = 0, n = messages.size(); i < n; i++)
                        entries.add(new DeleteMessageBatchRequestEntry()
                            .withId(Integer.toString(i))
                            .withReceiptHandle(messages.get(i)
                                .getReceiptHandle()));
                    batchRequest.setEntries(entries);

                    final DeleteMessageBatchResult batchResult = sqsClient
                        .deleteMessageBatch(batchRequest);
                    consumedCount.addAndGet(batchResult.getSuccessful().size());
                }
            }
        } catch (Exception e) {
            //
        }
    }
}
```

```
        * 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 [Pemantauantab](#) untuk antrian Anda atau pada [CloudWatch konsol](#).

 Note

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

Sumber daya Amazon SQS

Berikut ini adalah tabel yang mencantumkan daftar sumber daya terkait yang akan berguna saat Anda bekerja dengan layanan ini.

Sumber Daya	Deskripsi
Referensi Amazon Simple Queue Service	Deskripsi tindakan, parameter, dan jenis data serta daftar kesalahan yang dikembalikan layanan.
Amazon SQS diAWS CLIReferensi Perintah	Deskripsi dariAWS CLIperintah 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 untuk membahas pertanyaan teknis terkait Amazon SQS.
AWSInformasi Premium Support	Halaman web utama untuk informasi tentangAWSSupport Premium, saluran dukungan respons cepat langsung untuk membantu Anda membangun dan menjalankan aplikasiAWSlayanan infrastruktur.

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 menambahkan tindakan baru yang memungkinkan Anda mencantumkan tugas pergerakan pesan terbaru (hingga 10) di bawah antrian sumber tertentu. Tindakan ini dikaitkan dengan operasi API <code>ListMessageMoveTasks</code> .	Juni 7, 2023
Penggerak ulang antrian huruf mati menggunakan API	Konfigurasi 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 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. [listDeadLetterSourceQueues](#)

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 Lambdapemicu 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 `dataMessageSystemAttributeValue`.

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 menggunakan pola pesan umum seperti respon permintaan. Anda dapat menggunakan Klien Antrian Sementara untuk membuat antrian 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

Glosarium AWS

Lihat terminologi AWS terbaru di [AWS glosarium](#) dalam Referensi Glosarium AWS.

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.