aws

Panduan Pengguna

Amazon EventBridge

Amazon EventBridge: Panduan Pengguna

Table of Contents

Apa itu Amazon EventBridge? 1
CloudWatch Acara 2
Penyiapan dan prasyarat 3
Daftar Akun AWS 3
Membuat pengguna administratif 4
Masuk ke EventBridge konsol Amazon 5
Kredensial akun 5
Atur AWS Command Line Interface 6
Titik Akhir Regional 6
Memulai 7
Buat aturan 7
Bus Acara 10
Bagaimana bus acara bekerja 11
Konsep bus acara 13
Bus peristiwa 13
Peristiwa 14
Sumber kejadian 15
Aturan 15
Target 16
Fitur lanjutan 16
Membuat bus peristiwa 18
Memperbarui bus acara 19
Memperbarui izin bus acara 19
Memperbarui arsip 20
Memulai atau menghentikan penemuan skema 21
Memperbarui tag 21
Menghapus bus acara 22
Izin untuk bus peristiwa 22
Mengelola izin bus peristiwa 23
Contoh kebijakan: Kirim peristiwa ke bus default di akun yang berbeda 26
Contoh kebijakan: Kirim peristiwa ke bus kustomisasi di akun yang berbeda 26
Contoh kebijakan: Kirim peristiwa ke bus peristiwa di akun yang sama 27
Contoh kebijakan: Kirim peristiwa ke akun yang sama dan batasi pembaruan 28

Contoh kebijakan: Kirim peristiwa hanya dari aturan khusus ke bus di Wilayah yang berbeda 29
Contoh kebijakan: Kirim peristiwa hanya dari Wilayah tertentu ke Wilayah yang berbeda 30
Contoh kebijakan: Tolak pengiriman peristiwa dari Daerah tertentu 30
Menghasilkan template dari bus acara 31
Pertimbangan saat menggunakan template yang dihasilkan 33
Peristiwa 34
Referensi struktur acara 35
Acara kustom minimum yang valid 37
Menambahkan acara dengan PutEvents 37
Menangani kegagalan dengan `PutEvents` 39
Mengirim acara menggunakan AWS CLI 41
Menghitung ukuran entri acara 43
Acara dari AWS layanan 44
Pengiriman acara layanan 44
Acara melalui CloudTrail 45
Layanan yang menghasilkan acara 47
Acara manajemen 56
EventBridge acara 84
Menerima peristiwa dari mitra SaaS 91
Integrasi mitra SaaS yang didukung 91
Mengkonfigurasi EventBridge 94
Buat aturan untuk peristiwa mitra Saas 95
Menerima acara menggunakan URL fungsi Lambda 97
Menerima acara dari Salesforce 107
Pengiriman acara debugging 111
Menggunakan antrean surat mati 112
Pola peristiwa 117
Membuat pola acara 118
Mencocokkan nilai acara 119
Pertimbangan saat membuat pola acara 119
Operasi perbandingan untuk digunakan dalam pola acara 121
Contoh peristiwa dan pola peristiwa 123
Pencocokan bidang 123
Pencocokan nilai 124
Nilai null dan string kosong 126

Array 128
Penyaringan berbasis konten 129
Pencocokan prefiks 130
Pencocokan akhiran 130
Apa pun yang cocok 131
Pencocokan numerik 133
Pencocokan alamat IP 134
Pencocokan yang ada 134
quals-ignore-case Pencocokan E 135
Pencocokan menggunakan wildcard 135
Contoh kompleks dengan beberapa pencocokan 137
Contoh kompleks dengan `$or` pencocokan 138
Menguji pola acara 139
Praktik terbaik 143
Hindari menulis loop tak terbatas 143
Buat pola acara setepat mungkin 144
Cakupan pola acara Anda untuk memperhitungkan pembaruan sumber acara 146
Validasi pola acara 147
Aturan 148
Aturan terkelola 149
Membuat aturan yang bereaksi terhadap peristiwa 150
Buat aturan yang bereaksi terhadap peristiwa 150
Menggunakan EventBridge Scheduler 161
Mengatur peran eksekusi 161
Buat jadwal 162
Sumber daya terkait 166
Membuat aturan yang berjalan sesuai jadwal 167
Buat aturan yang berjalan sesuai jadwal 168
Ekspresi Cron 177
Ekspresi rate 182
Menonaktifkan atau menghapus aturan 184
Praktik terbaik 184
Tetapkan satu target untuk setiap aturan 184
Tetapkan izin aturan 185
Pantau kinerja aturan 185
MenggunakanAWS SAMtemplat 187

Templat gabungan ... 187
Templat terpisah ... 188
Menghasilkan templat aturan ... 189
Pertimbangan saat menggunakan template yang dihasilkan ... 191
Target ... 192
Target tersedia di EventBridge konsol ... 192
Parameter terget ... 193
Parameter jalur dinamis ... 194
Izin ... 195
EventBridge spesifik target ... 195
AWS Batch antrian pekerjaan ... 195
CloudWatch Grup log ... 196
CodeBuild proyek ... 196
Tugas Amazon ECS ... 196
Rencana Respons Manajer Insiden ... 196
Konfigurasi target ... 197
Tujuan API ... 198
API Gateway ... 220
AWS AppSync target ... 222
Koneksi ... 226
Bus acara lintas akun ... 230
Bus acara Lintas Wilayah ... 233
Bus acara akun yang sama ... 235
Transformasi masukan ... 237
Variabel yang telah ditetapkan ... 238
Contoh transformasi masukan ... 238
Mengubah input dengan menggunakan API EventBridge ... 241
Mengubah masukan dengan menggunakan AWS CloudFormation ... 242
Masalah Umum dengan mengubah input ... 242
Mengkonfigurasi transformator input ... 244
Menguji transformator input ... 247
Arsip dan putar ulang ... 252
Mengarsipkan peristiwa ... 253
Pemutaran ulang peristiwa yang diarsipkan ... 255
Pipa ... 257
Bagaimana Pipa bekerja ... 257

Konsep pipa ... 259
Pipa ... 259
Sumber ... 259
Filter ... 259
Pengayaan ... 260
Target ... 260
Izin untuk pipa ... 260
Izin DynamoDB ... 261
Izin Kinesis ... 262
Izin Amazon MQ ... 262
Izin MSK Amazon ... 263
Izin Apache Kafka yang dikelola sendiri ... 263
Izin Amazon SQS ... 265
Pengayaan dan izin target ... 265
Membuat pipa ... 265
Menentukan sumber ... 265
Mengkonfigurasi penyaringan ... 271
Mendefinisikan pengayaan ... 271
Mengkonfigurasi target ... 272
Mengkonfigurasi pengaturan pipa ... 273
Memvalidasi parameter konfigurasi ... 275
Memulai atau menghentikan pipa ... 275
Sumber ... 276
DynamoDB streams ... 277
Aliran kinesis ... 281
Pialang pesan Amazon MQ ... 284
Topik MSK Amazon ... 290
Aliran Apache Kafka ... 298
Antrean Amazon SQS ... 304
Penyaringan ... 308
Bidang pesan dan data ... 311
Memfilter pesan Amazon SQS ... 311
Memfilter pesan Kinesis dan DynamoDB ... 311
Memfilter Amazon MSK, Apache Kafka yang dikelola sendiri, dan pesan Amazon MQ ... 314
Perbedaan dengan Lambda ESM ... 315
Pengayaan ... 315

Menyaring peristiwa menggunakan pengayaan 316
Memanggil pengayaan 316
Target 317
Parameter terget 317
Izin 319
Memanggil target 319
EventBridge Pipa target spesifik 320
Batching dan konkurensi 320
Perilaku batching 320
Perilaku throughput dan konkurensi 323
Transformasi masukan 324
Variabel yang dicadangkan 326
Contoh transformasi masukan 327
Penguraian data tubuh implisit 328
Masalah umum dengan mengubah input 329
Kinerja pipa log 330
Cara kerja penebangan pipa 331
Menentukan tingkat log 332
Termasuk data eksekusi dalam log 334
Pelaporan kesalahan dalam catatan log 336
Langkah eksekusi pipa 337
Referensi skema log 340
Log & monitor 343
Penanganan kesalahan & pemecahan masalah 347
Coba lagi perilaku 347
Kesalahan pemanggilan dan perilaku coba lagi 347
Perilaku DLQ 349
Status kegagalan pipa 350
Kegagalan enkripsi khusus 350
Tutorial: Buat pipa yang memfilter acara 351
Prasyarat 351
Buat pipa 353
Konfirmasikan peristiwa filter pipa 355
Pembersihan sumber daya 356
Template untuk prasyarat 357
Menghasilkan template pipa 359

Sumber daya termasuk dalam template pipa 359
Pertimbangan saat menggunakan template yang dihasilkan 360
Menghasilkan CloudFormation template dari EventBridge Pipes 360
Titik Akhir Global 362
Waktu Pemulihan & Tujuan Titik Pemulihan 362
Replikasi peristiwa 363
Muatan peristiwa 363
Membuat titik akhir global 364
Untuk membuat titik akhir global menggunakan konsol 364
Untuk membuat titik akhir global menggunakan API 365
Untuk membuat titik akhir global menggunakanAWS CloudFormation 365
Bekerja dengan endpoint global dengan menggunakanAWSSDK 366
Wilayah yang Tersedia 366
Praktik terbaik 367
Mengaktifkan replikasi peristiwa 367
Mencegah throttling peristiwa 368
Menggunakan metrik pelanggan di pemeriksaan kondisi Amazon Route 53 368
templat AWS CloudFormation 368
AWS CloudFormationtemplate untuk menentukan pemeriksaan kondisi Route 53 368
Properti templat alarm CloudWatch 371
Sifat templat pemeriksaan kondisi 53 373
Skema 375
Penutupan nilai properti API registri skema 376
Menemukan skema 377
Registri skema 378
Membuat skema 379
Membuat skema dengan menggunakan templat 379
Edit templat skema secara langsung di konsol tersebut 381
Buat skema dari JSON dari peristiwa 382
Membuat skema dari peristiwa pada bus peristiwa 385
Pengikatan kode 387
AWSLayanan dan alat terkait 388
Antarmuka VPC Endpoint 389
Ketersediaan 389
Membuat Endpoint VPC untuk EventBridge 390
EventBridge Spesifikasi pipa 391

AWS X-Ray ... 392
Pengujian dengan AWS IATK ... 393
AWS Integrasi IATK ... 393
AWS CloudFormation ... 394
Sumber daya EventBridge ... 394
Menghasilkan definisi sumber daya ... 395
Mengelola acara CloudFormation tumpukan ... 395
Tutorial ... 397
Memulai tutorial ... 398
Arsip dan putar ulang peristiwa ... 399
Buat aplikasi sampel ... 404
Mengunduh pengikatan kode ... 409
Gunakan transformator input ... 411
Tutorial AWS ... 416
Log status grup Auto Scaling ... 417
Log AWS panggilan API ... 421
Log status instans Amazon EC2 ... 426
Operasi tingkat objek Log Amazon S3 ... 430
Mengirim kejadian ke aliran Kinesis menggunakan`aws.events` ... 435
Jadwalkan Snapshot Amazon EBS Otomatis ... 440
Kirim pemberitahuan saat objek S3 dibuat ... 443
Jadwalkan AWS Lambda fungsi ... 447
Tutorial SaaS ... 452
Buat koneksi keDatadog ... 453
Buat koneksi ke Salesforce ... 458
Buat koneksi keZendesk ... 463
Bekerja dengan AWS SDK ... 468
Contoh kode ... 470
Tindakan ... 474
`DeleteRule` ... 475
`DescribeRule` ... 477
`DisableRule` ... 480
`EnableRule` ... 483
`ListRuleNamesByTarget` ... 487
`ListRules` ... 490
`ListTargetsByRule` ... 493

PutEvents ... 496
PutRule ... 504
PutTargets ... 513
RemoveTargets ... 523
Skenario ... 527
Buat dan picu aturan ... 528
Memulai dengan aturan dan target ... 549
Contoh lintas layanan ... 609
Menggunakan peristiwa terjadwal untuk menginvokasi fungsi Lambda ... 609
Keamanan ... 612
Perlindungan data ... 613
Enkripsi saat tidak aktif ... 614
Enkripsi dalam transit ... 614
Kebijakan berbasis tanda ... 615
IAM ... 616
Autentikasi ... 616
Kontrol akses ... 618
Mengelola akses ... 619
Menggunakan kebijakan berbasis identitas (kebijakan IAM) ... 625
Menggunakan kebijakan berbasis sumber daya ... 643
Pencegahan Deputi Bingung Lintas Layanan ... 649
Kebijakan berbasis sumber daya untuk skema EventBridge ... 652
Referensi izin ... 656
Ketentuan kebijakan IAM ... 659
Menggunakan peran terkait layanan ... 677
CloudTrail log ... 684
Peristiwa data ... 685
Acara manajemen ... 687
Contoh acara ... 687
Event untuk tindakan Pipe ... 688
Validasi kepatuhan ... 691
Ketahanan ... 692
Keamanan infrastruktur ... 693
Analisis keamanan dan kelemahan ... 694
Memantau ... 695
EventBridge metrik ... 695

EventBridge PutEvents metrik 699
EventBridge PutPartnerEvents metrik 701
Dimensi untuk EventBridge metrik 702
Pemecahan Masalah 703
Aturan saya berjalan tetapi fungsi Lambda saya tidak dipanggil 703
Saya baru saja membuat atau memodifikasi aturan, tetapi tidak cocok dengan kejadian pengujian 705
Aturan saya tidak berjalan pada waktu yang saya tentukan di `ScheduleExpression` 706
Aturan saya tidak berjalan pada waktu yang saya harapkan 706
Aturan saya cocok dengan panggilan API layanan AWS global tetapi tidak berjalan 707
IAM role (IAM role) yang terkait dengan aturan saya diabaikan saat aturan berjalan 707
Aturan saya memiliki pola kejadian yang seharusnya cocok dengan sumber daya, tapi tidak ada kejadian yang cocok 707
Pengiriman kejadian saya ke target tertunda 708
Beberapa kejadian tidak pernah dikirimkan ke target saya 708
Aturan saya berjalan lebih dari sekali dalam menanggapi satu kejadian 708
Mencegah loop tak terbatas 708
Kejadian saya tidak dikirim ke antrean Amazon SQS target 709
Aturan saya berjalan, tapi saya tidak melihat pesan yang diterbitkan ke topik Amazon SNS saya 709
Topik Amazon SNS saya masih memiliki izin EventBridge bahkan setelah saya menghapus aturan yang terkait dengan topik Amazon SNS 711
Kunci kondisi IAM mana yang dapat saya gunakan? EventBridge 711
Bagaimana saya bisa tahu kapan EventBridge aturan dilanggar? 712
Kuota 713
EventBridge kuota 713
PutPartnerEvents kuota 721
Kuota Registri Skema 722
Kuota pipa 723
Tag 725
Riwayat Dokumen 727
..... dccxxxiv

Apa itu Amazon EventBridge?

EventBridge adalah layanan tanpa server yang menggunakan peristiwa untuk menghubungkan komponen aplikasi bersama-sama, sehingga memudahkan Anda untuk membangun aplikasi berbasis peristiwa yang dapat diskalakan. Arsitektur berbasis peristiwa adalah gaya membangun sistem perangkat lunak yang digabungkan secara longgar yang bekerja sama dengan memancarkan dan menanggapi peristiwa. Arsitektur berbasis peristiwa dapat membantu Anda meningkatkan kelincahan dan membangun aplikasi yang andal dan dapat diskalakan.

Gunakan EventBridge untuk merutekan acara dari sumber seperti aplikasi rumahan, AWS layanan, dan perangkat lunak pihak ketiga ke aplikasi konsumen di seluruh organisasi Anda. EventBridge menyediakan cara sederhana dan konsisten untuk menelan, memfilter, mengubah, dan menyampaikan acara sehingga Anda dapat membangun aplikasi dengan cepat.

Video berikut memberikan pengantar singkat tentang fitur-fitur Amazon EventBridge:

EventBridge mencakup dua cara untuk memproses acara: bus acara dan pipa.

- Bus acara adalah router yang menerima acara dan mengirimkannya ke nol atau lebih target. Bus acara sangat cocok untuk merutekan acara dari banyak sumber ke banyak target, dengan transformasi opsional acara sebelum pengiriman ke target.

  Video berikut memberikan ikhtisar tingkat tinggi tentang bus acara:

- EventBridge Pipa Pipa dimaksudkan untuk point-to-point integrasi; setiap pipa menerima peristiwa dari satu sumber untuk pemrosesan dan pengiriman ke satu target. Pipa juga mencakup dukungan untuk transformasi lanjutan dan pengayaan peristiwa sebelum pengiriman ke target.

Pipa dan bus acara sering digunakan bersama. Kasus penggunaan yang umum adalah membuat pipa dengan bus acara sebagai targetnya; pipa mengirimkan peristiwa ke bus acara, yang kemudian mengirimkan peristiwa tersebut ke beberapa target. Misalnya, Anda dapat membuat pipa dengan aliran DynamoDB untuk sumber, dan bus acara sebagai target. Pipa menerima peristiwa dari aliran DynamoDB dan mengirimkannya ke bus acara, yang kemudian mengirimkannya ke beberapa target sesuai dengan aturan yang telah Anda tentukan di bus acara.

EventBridge adalah evolusi dari Amazon CloudWatch Events

EventBridge Sebelumnya bernama Amazon CloudWatch Events. Bus acara default dan aturan yang Anda buat di CloudWatch Acara juga ditampilkan di EventBridge konsol. EventBridge menggunakan CloudWatch Events API yang sama, sehingga kode Anda yang menggunakan CloudWatch Events API tetap sama.

EventBridge dibangun di atas kemampuan CloudWatch Acara dengan fitur seperti acara mitra, Schema Registry, dan EventBridge Pipes. Fitur baru yang EventBridge ditambahkan ke tidak ditambahkan ke CloudWatch Acara. Untuk informasi selengkapnya, lihat ???.

Semua fitur yang biasa Anda gunakan di CloudWatch Acara juga ada EventBridge, termasuk:

- ???
- ???
- ???
- ???

EventBridge fitur yang membangun dan memperluas kemampuan acara meliputi:

- ???
- ???
- ???
- ???

EventBridge Penyiapan dan prasyarat Amazon

Untuk menggunakan Amazon EventBridge, Anda memerlukan AWS akun. Akun Anda memungkinkan Anda menggunakan layanan seperti Amazon EC2 untuk menghasilkan peristiwa yang dapat Anda lihat di EventBridge konsol. Anda juga dapat menginstal dan mengkonfigurasi AWS Command Line Interface (AWS CLI) untuk menggunakan antarmuka baris perintah untuk melihat peristiwa.

Topik

- Daftar Akun AWS
- Membuat pengguna administratif
- Masuk ke EventBridge konsol Amazon
- Kredensial akun
- Atur AWS Command Line Interface
- Titik Akhir Regional

Daftar 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 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 mengirimi Anda email konfirmasi 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 Anda mendaftarAkun AWS, amankanPengguna root akun AWS, aktifkanAWS IAM Identity Center, dan buat 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. Pada halaman berikutnya, masukkan kata sandi Anda.

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

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

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

Masuk sebagai pengguna administratif

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

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

Masuk ke EventBridge konsol Amazon

Untuk masuk ke EventBridge konsol Amazon

- Masuk ke AWS Management Console dan buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.

Kredensial akun

Meskipun Anda dapat menggunakan kredensi pengguna root Anda untuk mengakses EventBridge, kami sarankan Anda menggunakan akun AWS Identity and Access Management (IAM) sebagai gantinya. Jika Anda menggunakan akun IAM untuk mengakses EventBridge, Anda harus memiliki izin berikut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "events:*"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:events:*:*:*"
    },
    {
      "Action": [
        "iam:PassRole"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "StringLike": {
```

```
          "iam:PassedToService": "events.amazonaws.com"
        }
      }
    }
  ]
}
```

Untuk informasi selengkapnya, lihat Autentikasi.

Atur AWS Command Line Interface

Anda dapat menggunakan AWS CLI untuk melakukan EventBridge operasi.

Untuk informasi tentang cara memasang dan mengonfigurasi AWS CLI, lihat Menyiapkan AWS Command Line Interface di AWS Command Line Interface Panduan Pengguna.

Titik Akhir Regional

Anda harus mengaktifkan titik akhir regional default untuk digunakan EventBridge. Untuk informasi selengkapnya, lihat Mengaktifkan dan menonaktifkan AWS STS di AWS Wilayah dalam Panduan Pengguna IAM.

Memulai dengan Amazon EventBridge

Dasarnya EventBridge adalah membuat aturan yang merutekan peristiwa ke target. Di bagian ini, Anda akan membuat aturan dasar. Untuk tutorial mengenai skenario dan target tertentu, lihat Tutorial Amazon EventBridge.

Buat aturan di Amazon EventBridge

Untuk membuat aturan untuk acara, Anda menentukan tindakan yang akan diambil saat EventBridge menerima peristiwa yang cocok dengan pola acara dalam aturan. Saat acara cocok, EventBridge kirimkan acara ke target yang ditentukan dan memicu tindakan yang ditentukan dalam aturan.

Ketika AWS layanan di AWS akun Anda memancarkan acara, itu selalu masuk ke bus acara default untuk akun Anda. Untuk menulis aturan yang cocok dengan acara dari AWS layanan di akun Anda, Anda harus mengaitkannya dengan bus acara default.

Untuk membuat aturan untuk AWS layanan

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan.

   Aturan tidak boleh memiliki nama yang sama dengan aturan lain di Wilayah yang sama dan di bus kejadian yang sama.
5. Untuk bus acara, pilih bus acara yang ingin Anda kaitkan dengan aturan ini. Jika Anda ingin aturan ini cocok dengan acara yang berasal dari akun Anda, pilih bus acara AWS default. Saat layanan AWS di akun Anda menghasilkan kejadian, layanan tersebut akan selalu masuk ke bus kejadian default akun Anda.
6. Untuk Tipe aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.
8. Untuk Sumber peristiwa, pilih Layanan AWS .
9. (Opsional) Untuk contoh acara, pilih jenis acara.
10. Untuk pola Acara, lakukan salah satu hal berikut:

- Untuk menggunakan templat untuk membuat pola acara Anda, pilih Formulir pola acara dan pilih Sumber acara dan jenis Acara. Jika Anda memilih Semua Acara sebagai jenis acara, semua peristiwa yang dipancarkan oleh AWS layanan ini akan cocok dengan aturan.

  Untuk menyesuaikan template, pilih Pola kustom (editor JSON) dan buat perubahan Anda.
- Untuk menggunakan pola acara khusus, pilih Pola kustom (editor JSON) dan buat pola acara Anda.

11. Pilih Berikutnya.
12. Untuk Jenis target, pilih Layanan AWS .
13. Untuk Pilih target, pilih AWS layanan yang ingin Anda kirimi informasinya saat EventBridge mendeteksi peristiwa yang cocok dengan pola acara.
14. Bidang yang ditampilkan bervariasi tergantung pada layanan yang Anda pilih. Masukkan informasi khusus untuk jenis target ini sesuai kebutuhan.
15. Untuk banyak jenis target, EventBridge perlu izin untuk mengirim acara ke target. Dalam kasus ini, EventBridge dapat membuat peran IAM yang diperlukan agar aturan Anda berjalan. Lakukan salah satu dari langkah berikut ini:

    - Untuk membuat IAM role secara otomatis, pilih Buat peran baru untuk sumber daya khusus ini.
    - Untuk menggunakan peran IAM yang Anda buat sebelumnya, pilih Gunakan peran yang ada dan pilih peran yang ada dari daftar drop-down.
16. (Opsional) Untuk pengaturan tambahan, lakukan hal berikut:

    a. Untuk Masa peristiwa maksimal, masukkan nilai antara satu menit (00:01) dan 24 jam (24:00).

    b. Untuk Upaya coba lagi, masukkan angka antara 0 dan 185.

    c. Untuk antrian Dead-letter, pilih apakah akan menggunakan antrean Amazon SQS standar sebagai antrian huruf mati. EventBridge mengirimkan peristiwa yang cocok dengan aturan ini ke antrian huruf mati jika tidak berhasil dikirim ke target. Lakukan salah satu hal berikut:

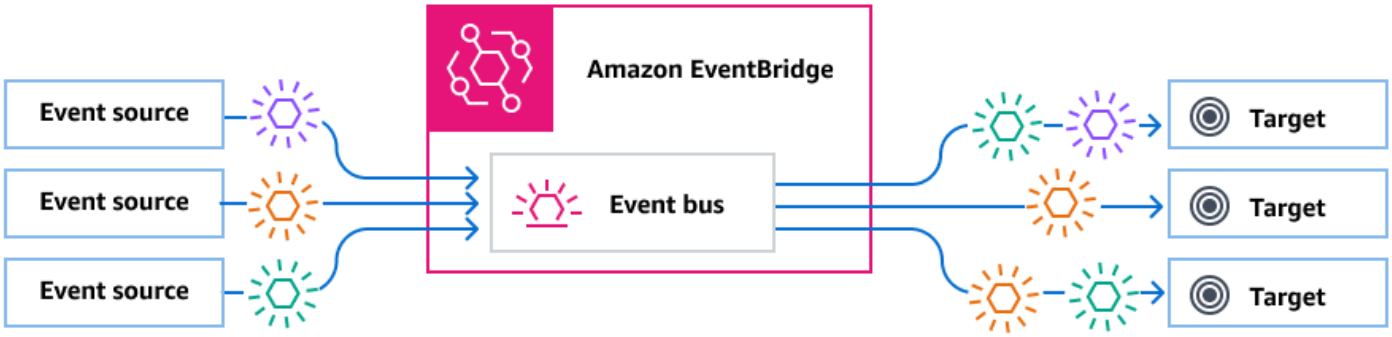
        - Pilih Tidak ada untuk tidak menggunakan antrean surat mati.
        - Pilih Pilihan antrean Amazon SQS di akun AWS saat ini untuk digunakan sebagai antrean surat mati kemudian pilih antrean yang akan digunakan dari daftar menurun.
        - Pilih Pilih antrean Amazon SQS di AWS akun lain sebagai antrian huruf mati dan kemudian masukkan ARN antrian yang akan digunakan. Anda harus melampirkan kebijakan berbasis sumber daya ke antrian yang memberikan EventBridge izin untuk

mengirim pesan ke antrean tersebut. Untuk informasi selengkapnya, lihat Memberikan izin untuk antrean surat mati.

17. (Opsional) Pilih Tambahkan target lain untuk menambahkan target lain untuk aturan ini.
18. Pilih Berikutnya.
19. (Opsional) Masukkan satu atau lebih tanda untuk aturan. Untuk informasi selengkapnya, lihat EventBridge Tag Amazon.
20. Pilih Berikutnya.
21. Tinjau detail aturan dan pilih Buat aturan.

Bus EventBridge Acara Amazon

Bus acara adalah router yang menerima acara dan mengirimkannya ke nol atau lebih tujuan, atau target. Bus acara sangat cocok untuk merutekan acara dari banyak sumber ke banyak target, dengan transformasi opsional acara sebelum pengiriman ke target.



Aturan yang terkait dengan bus peristiwa mengevaluasi peristiwa saat datang. Setiap aturan memeriksa apakah suatu peristiwa cocok dengan pola aturan. Jika acara tidak cocok, EventBridge kirimkan acara

Anda menghubungkan aturan dengan bus peristiwa tertentu, sehingga aturan hanya berlaku untuk peristiwa yang diterima oleh bus peristiwa.

> **Note**
>
> Anda juga dapat memproses acara menggunakan EventBridge Pipes. EventBridge Pipa dimaksudkan untuk point-to-point integrasi; setiap pipa menerima peristiwa dari satu sumber untuk pemrosesan dan pengiriman ke satu target. Pipa juga mencakup dukungan untuk transformasi lanjutan dan pengayaan peristiwa sebelum pengiriman ke target. Untuk informasi selengkapnya, lihat ???.

Topik

- Bagaimana bus acara bekerja
- Konsep Bus EventBridge Acara Amazon
- Membuat bus EventBridge acara Amazon
- Memperbarui bus EventBridge acara Amazon

- Menghapus bus EventBridge acara Amazon
- Izin untuk busEventBridge peristiwa Amazon
- Buat AWS CloudFormation template dari bus EventBridge acara Amazon

Bagaimana bus acara bekerja

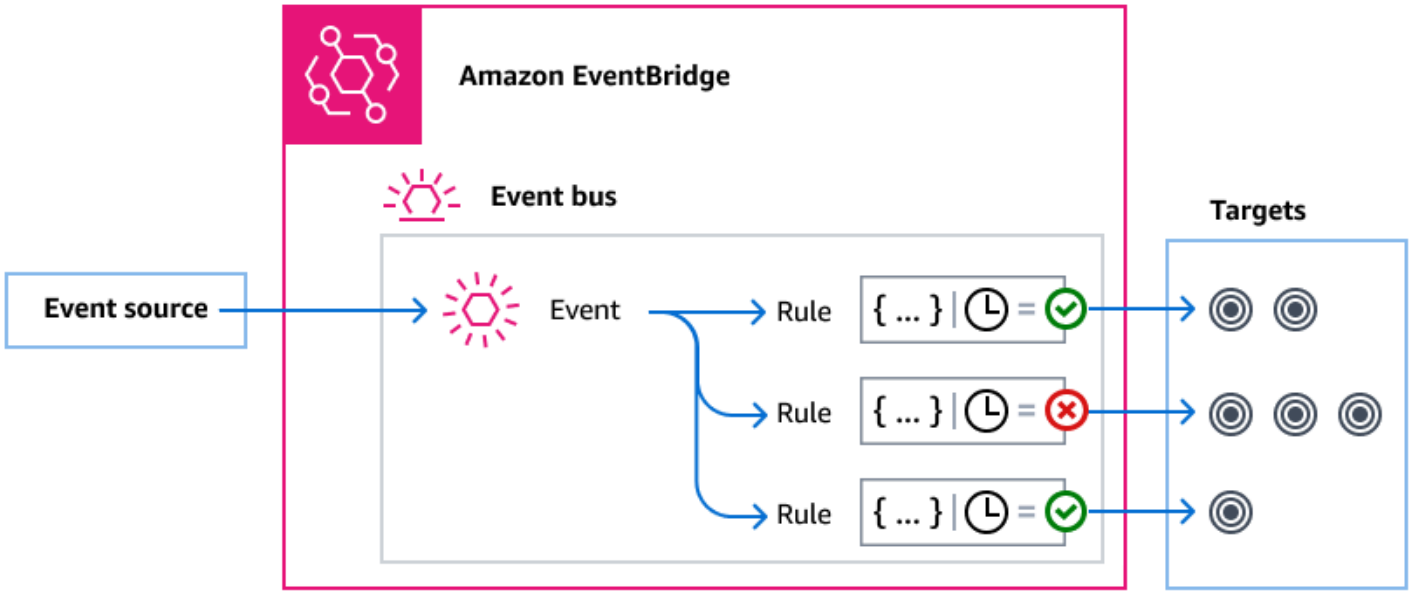
Bus acara memungkinkan Anda untuk merutekan acara dari berbagai sumber ke beberapa tujuan, atau target.

Pada tingkat tinggi, inilah cara kerjanya:

1. Sumber acara, yang dapat berupa AWS layanan, aplikasi kustom Anda sendiri, atau penyedia SaaS, mengirimkan acara ke bus acara.
2. EventBridge kemudian mengevaluasi peristiwa terhadap setiap aturan yang ditentukan untuk bus acara tersebut.

   Untuk setiap peristiwa yang cocok dengan aturan, EventBridge lalu kirimkan acara ke target yang ditentukan untuk aturan itu. Secara opsional, sebagai bagian dari aturan, Anda juga dapat menentukan bagaimana EventBridge harus mengubah acara sebelum mengirimnya ke target.

   Sebuah peristiwa mungkin cocok dengan beberapa aturan, dan setiap aturan dapat menentukan hingga lima target. (Suatu peristiwa mungkin tidak cocok dengan aturan apa pun, dalam hal ini tidak EventBridge mengambil tindakan.)



Pertimbangkan contoh menggunakan bus acara EventBridge default, yang secara otomatis menerima acara dari AWS layanan:

1. Anda membuat aturan pada bus acara default untuk `EC2 Instance State-change Notification` acara tersebut:
   - Anda menentukan bahwa aturan cocok dengan peristiwa di mana instans Amazon EC2 telah berubah menjadi. `state running`

     Anda melakukan ini dengan menentukan JSON yang mendefinisikan atribut dan nilai suatu peristiwa harus cocok untuk memicu aturan. Ini disebut pola peristiwa.

     ```
     {
       "source": ["aws.ec2"],
       "detail-type": ["EC2 Instance State-change Notification"],
         "detail": {
           "state": ["running"]
       }
     }
     ```

   - Anda menentukan target aturan untuk menjadi fungsi Lambda yang diberikan.
2. Setiap kali instans Amazon EC2 berubah status, Amazon EC2 (sumber peristiwa) secara otomatis mengirimkan peristiwa tersebut ke bus peristiwa default.
3. EventBridge mengevaluasi semua peristiwa yang dikirim ke bus acara default terhadap aturan yang Anda buat.

Jika acara cocok dengan aturan Anda (yaitu, jika peristiwa tersebut merupakan instans Amazon EC2 yang mengubah status menjadi`running`), EventBridge kirimkan acara ke target yang ditentukan. Dalam hal ini, itulah fungsi Lambda.

Video berikut menjelaskan apa itu bus acara dan apa yang mereka lakukan: Apa itu bus acara

Video berikut mencakup bus acara yang berbeda dan kapan menggunakannya: Perbedaan antara bus acara

Konsep Bus EventBridge Acara Amazon

Berikut adalah melihat lebih dekat komponen utama dari arsitektur berbasis acara yang dibangun di atas bus acara.

Bus peristiwa

Bus acara adalah router yang menerima acara dan mengirimkannya ke nol atau lebih tujuan, atau target. Gunakan bus acara saat Anda perlu merutekan acara dari banyak sumber ke banyak target, dengan transformasi acara opsional sebelum pengiriman ke target.

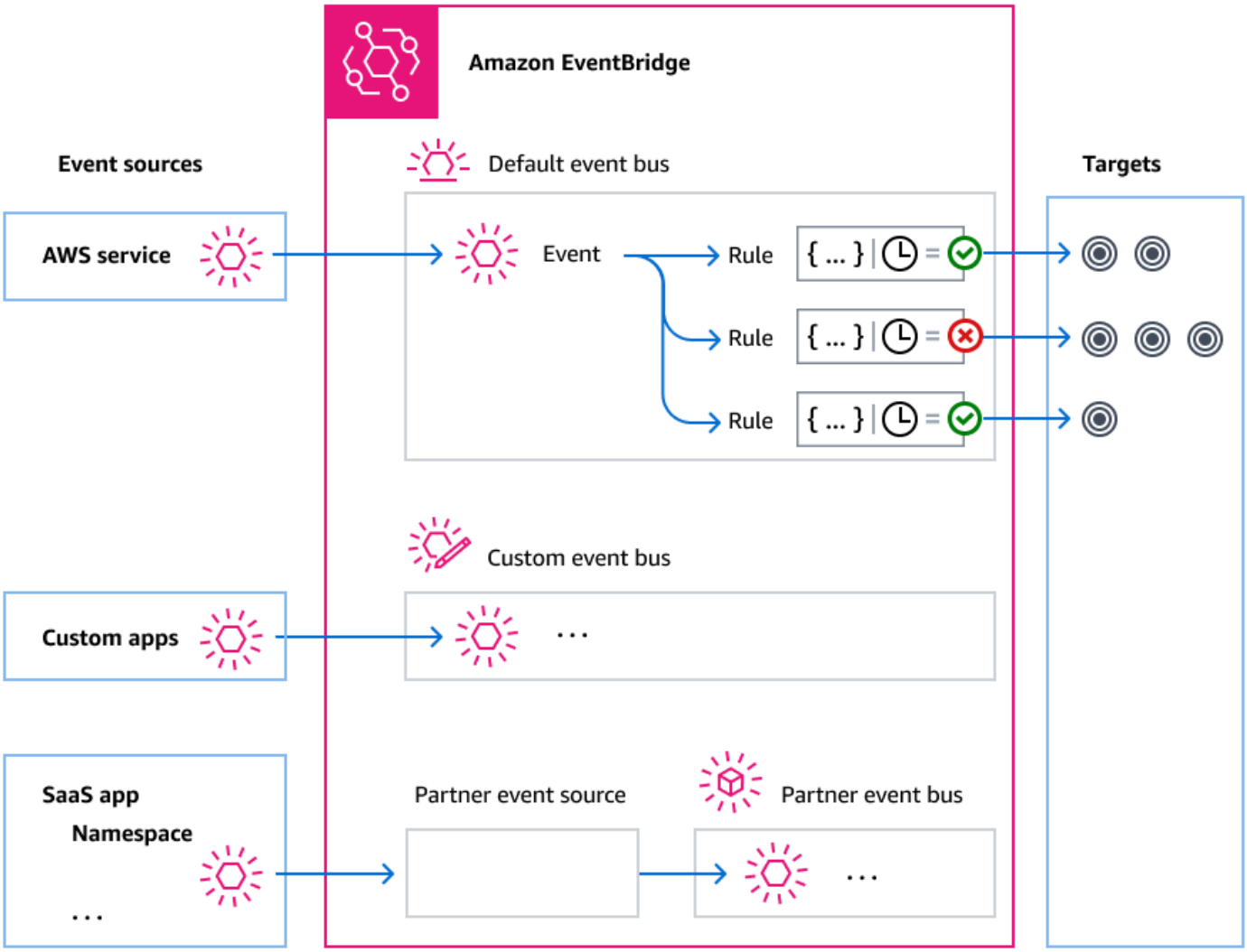
Akun Anda menyertakan bus acara default yang secara otomatis menerima acara dari AWS layanan. Anda juga dapat:

- Buat bus acara tambahan, yang disebut bus acara khusus, dan tentukan acara mana yang mereka terima.
- Buat bus acara mitra, yang menerima acara dari mitra SaaS.

Kasus penggunaan umum untuk bus acara meliputi:

- Menggunakan bus acara sebagai broker antara beban kerja, layanan, atau sistem yang berbeda.
- Menggunakan beberapa bus acara dalam aplikasi Anda untuk membagi lalu lintas acara. Misalnya, membuat bus untuk memproses acara yang berisi informasi identifikasi pribadi (PII), dan bus lain untuk acara yang tidak.

- Menggabungkan acara dengan mengirimkan acara dari beberapa bus acara ke bus acara terpusat. Bus terpusat ini dapat berada di akun yang sama dengan bus lainnya, tetapi juga dapat berada di akun atau Wilayah yang berbeda.



Peristiwa

Paling sederhana, sebuah EventBridge peristiwa adalah objek JSON yang dikirim ke bus acara atau pipa.

Dalam konteks arsitektur berbasis peristiwa (EDA), suatu peristiwa sering mewakili indikator perubahan sumber daya atau lingkungan.

Untuk informasi selengkapnya, lihat ???.

Sumber kejadian

EventBridge dapat menerima acara dari sumber acara termasuk:

- AWS layanan
- Aplikasi khusus
- Perangkat lunak sebagai mitra layanan (SaaS)

Aturan

Aturan menerima peristiwa yang masuk dan mengirimkannya sesuai dengan target untuk diproses. Anda dapat menentukan bagaimana setiap aturan memanggil target mereka berdasarkan:

- Pola acara, yang berisi satu atau lebih filter untuk mencocokkan acara. Pola acara dapat mencakup filter yang cocok pada:
  - Metadata peristiwa — Data tentang peristiwa, seperti sumber acara, atau akun atau Wilayah tempat acara tersebut berasal.
  - Data peristiwa — Properti acara itu sendiri. Properti ini bervariasi sesuai dengan acara.
  - Konten acara - Nilai properti aktual dari data peristiwa.
- Jadwal untuk memanggil target secara berkala.

  Anda dapat menentukan aturan terjadwal dalam EventBridge, atau dengan menggunakan EventBridge Scheduler.

  > **Note**
  >
  > EventBridge menawarkan Amazon EventBridge Scheduler, penjadwal tanpa server yang memungkinkan Anda membuat, menjalankan, dan mengelola tugas dari satu layanan terpusat dan terkelola. EventBridge Scheduler sangat dapat disesuaikan, dan menawarkan skalabilitas yang ditingkatkan dibandingkan aturan EventBridge terjadwal, dengan serangkaian operasi dan layanan API target yang lebih luas. AWS Kami menyarankan Anda menggunakan EventBridge Scheduler untuk memanggil target pada jadwal. Untuk informasi selengkapnya, lihat ???.

Setiap aturan ditetapkan untuk bus acara tertentu, dan hanya berlaku untuk acara di bus acara tersebut.

Satu aturan dapat mengirim acara hingga lima target.

Secara default, Anda dapat mengonfigurasi hingga 300 aturan per bus acara. Kuota ini dapat dinaikkan menjadi ribuan aturan di konsol Service Quotas. Karena batas aturan berlaku untuk setiap bus, jika Anda memerlukan lebih banyak aturan, Anda dapat membuat bus acara khusus tambahan di akun Anda.

Anda dapat menyesuaikan bagaimana peristiwa diterima di akun Anda dengan membuat bus peristiwa dengan izin yang berbeda untuk layanan yang berbeda.

Untuk menyesuaikan struktur atau tanggal suatu peristiwa sebelum EventBridge meneruskannya ke target, gunakan transformator input untuk mengedit informasi sebelum masuk ke target.

Untuk informasi selengkapnya, lihat ???.

Target

Target adalah sumber daya atau titik akhir yang EventBridge mengirimkan peristiwa saat acara cocok dengan pola acara yang ditentukan untuk aturan.

Target dapat menerima beberapa acara dari beberapa bus acara.

Untuk informasi selengkapnya, lihat ???.

Fitur canggih untuk bus acara

EventBridge mencakup fitur-fitur berikut untuk membantu Anda mengembangkan, mengelola, dan menggunakan bus acara.

Menggunakan tujuan API untuk mengaktifkan panggilan REST API antar layanan

EventBridge Tujuan API adalah titik akhir HTTP yang dapat Anda tetapkan sebagai target aturan, dengan cara yang sama seperti Anda akan mengirim data peristiwa ke AWS layanan atau sumber daya. Dengan menggunakan tujuan API, Anda dapat menggunakan panggilan API untuk merutekan peristiwa antar AWS layanan, aplikasi SaaS terintegrasi, dan aplikasi Anda di luar. AWS Saat Anda membuat tujuan API, Anda menentukan koneksi yang akan digunakan untuknya. Setiap koneksi termasuk detail tentang jenis otorisasi dan parameter yang akan digunakan untuk mengotorisasi dengan titik akhir tujuan API.

Mengarsipkan dan memutar ulang acara untuk membantu pembangunan dan pemulihan bencana

Anda dapat mengarsipkan, atau menyimpan, peristiwa dan kemudian memutarnya kembali dari arsip di lain waktu. Pengarsipan berguna untuk:

- Menguji aplikasi karena Anda memiliki penyimpanan acara untuk digunakan daripada harus menunggu acara baru.
- Menghidrasi layanan baru saat pertama kali online.
- Menambahkan lebih banyak daya tahan ke aplikasi berbasis acara Anda.

Menggunakan Schema Registry untuk memulai pembuatan pola acara

Ketika Anda membangun aplikasi tanpa server yang menggunakan EventBridge, akan sangat membantu untuk mengetahui struktur peristiwa biasa tanpa harus menghasilkan acara. Struktur acara dijelaskan dalam skema, yang tersedia untuk semua acara yang dihasilkan oleh AWS layanan di EventBridge.

Untuk acara yang tidak berasal dari AWS layanan, Anda dapat:

- Buat atau unggah skema khusus.
- Gunakan Schema Discovery untuk secara EventBridge otomatis membuat skema untuk acara yang dikirim ke bus acara.

Setelah Anda memiliki skema untuk peristiwa, Anda dapat mengunduh pengikatan kode untuk bahasa pemrograman populer.

Mengelola sumber daya dan akses dengan kebijakan

Untuk mengatur AWS sumber daya atau melacak biaya EventBridge, Anda dapat menetapkan label kustom, atau tag, ke AWS sumber daya. Dengan menggunakan kebijakan berbasis tag, Anda dapat mengontrol sumber daya apa yang dapat dan tidak dapat dilakukan di dalamnya EventBridge.

Selain kebijakan berbasis tag, EventBridge mendukung kebijakan berbasis identitas dan sumber daya untuk mengontrol akses ke. EventBridge Gunakan kebijakan berbasis identitas untuk mengontrol izin grup, peran, atau pengguna. Gunakan kebijakan berbasis sumber daya untuk memberikan izin khusus ke setiap sumber daya, seperti fungsi Lambda atau topik Amazon SNS.

Membuat bus EventBridge acara Amazon

Anda dapat membuat khusus bus peristiwa untuk menerima peristiwa dari aplikasi Anda. Aplikasi Anda juga dapat mengirim peristiwa ke bus peristiwa default. Bila Anda membuat bus peristiwa, Anda dapat melampirkan kebijakan berbasis sumber daya untuk memberikan izin ke akun lain. Kemudian akun lain dapat mengirim peristiwa ke bus peristiwa di akun saat ini.

Video berikut membahas pembuatan bus acara: Membuat bus acara

Untuk membuat bus peristiwa kustomisasi

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Bus peristiwa.
3. Pilih Buat bus peristiwa.
4. Masukkan nama untuk bus peristiwa baru.
5. Konfigurasikan bus acara:

   - Tentukan kebijakan berbasis sumber daya dengan melakukan salah satu hal berikut:
     - Masukkan kebijakan yang mencakup izin untuk memberikan bus peristiwa. Anda dapat menyisipkan kebijakan dari sumber lain atau memasukkan JSON untuk kebijakan. Anda dapat menggunakan salah satu contoh kebijakan dan memodifikasinya untuk lingkungan Anda.
     - Jika ingin menggunakan templat untuk kebijakan, pilih Muat template. Ubah kebijakan yang sesuai dengan lingkungan Anda, termasuk menambahkan tindakan yang mengizinkan utama dalam kebijakan untuk menggunakannya.

     Untuk informasi selengkapnya tentang pemberian izin ke bus acara melalui kebijakan berbasis sumber daya, lihat. ???
   - Aktifkan arsip (opsional)

     Anda dapat membuat arsip acara sehingga Anda dapat dengan mudah memutar ulang mereka di lain waktu. Misalnya, Anda mungkin ingin memutar ulang peristiwa untuk melakukan pemulihan dari kesalahan atau membutuhkan validasi fungsi baru aplikasi Anda. Lihat informasi yang lebih lengkap di ???

     a. Di bawah Arsip, pilih Diaktifkan.

     b. Tentukan nama dan deskripsi untuk arsip.

- Aktifkan penemuan skema (opsional)

  Aktifkan penemuan skema untuk secara EventBridge otomatis menyimpulkan skema langsung dari peristiwa yang berjalan di bus acara ini. Lihat informasi yang lebih lengkap di ???

  a. Di bawah Penemuan skema, pilih Diaktifkan.

- Tentukan tag (opsional)

  Tag adalah label atribut kustom yang Anda tetapkan ke AWS sumber daya. Gunakan tag untuk mengidentifikasi dan mengatur AWS sumber daya Anda. Banyak AWS layanan mendukung penandaan, sehingga Anda dapat menetapkan tag yang sama ke sumber daya dari layanan yang berbeda untuk menunjukkan bahwa sumber daya terkait. Lihat informasi yang lebih lengkap di ???

  a. Di bawah Tag, pilih Tambahkan tag baru.

  b. Tentukan kunci dan, secara opsional, nilai untuk tag baru.

6. Pilih Buat.

Memperbarui bus EventBridge acara Amazon

Anda dapat memperbarui konfigurasi bus acara setelah Anda membuatnya. Ini termasuk bus acara default, yang EventBridge dibuat di akun Anda secara otomatis.

Topik

- Memperbarui izin pada bus acara
- Menambahkan atau menghapus arsip di bus acara
- Memulai atau menghentikan penemuan skema di bus acara
- Menambahkan atau menghapus tag pada bus acara

Memperbarui izin pada bus acara

Anda dapat memberikan izin tambahan untuk bus peristiwa dengan melampirkan kebijakan berbasis sumber daya. Untuk petunjuk terperinci tentang memperbarui izin yang diberikan bus acara, lihat Mengelola izin bus acara.

Menambahkan atau menghapus arsip di bus acara

Arsip memungkinkan Anda untuk menangkap peristiwa sehingga Anda dapat dengan mudah memutar ulang mereka di lain waktu. Misalnya, Anda mungkin ingin memutar ulang peristiwa untuk melakukan pemulihan dari kesalahan atau membutuhkan validasi fungsi baru aplikasi Anda. Untuk informasi selengkapnya, lihat EventBridge arsip dan putar ulang.

Untuk menambah atau menghapus arsip dari bus acara menggunakan EventBridge konsol

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Bus peristiwa.
3. Pilih bus acara yang ingin Anda perbarui.
4. Pada halaman detail bus acara, pilih tab Arsip.
5. Lakukan salah satu dari cara berikut:
   - Untuk menambahkan arsip:
     a. Pilih Buat arsip.
     b. Tentukan atribut untuk arsip.
     c. Pilih Berikutnya.
     d. Pilih pola acara untuk diterapkan ke acara untuk arsip.
     e. Pilih Buat arsip.
   - Untuk menghapus arsip:
     a. Untuk tag yang ingin Anda hapus, pilih Hapus.
     b. Masukkan nama arsip, dan pilih Hapus.

        Arsip dihapus secara permanen. Anda tidak dapat membatalkan operasi ini.

Untuk membuat atau menghapus arsip untuk bus acara menggunakan AWS CLI

- Untuk membuat arsip, gunakan create-archive.

  Untuk menghapus arsip secara permanen, gunakan hapus-arsip.

Memulai atau menghentikan penemuan skema di bus acara

Untuk informasi lebih lanjut tentang penemuan skema, lihat EventBridge skema.

Untuk memulai atau menghentikan penemuan skema pada bus acara menggunakan konsol EventBridge

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Bus peristiwa.
3. Pilih bus acara yang ingin Anda perbarui.
4. Lakukan salah satu dari cara berikut:
   - Untuk memulai penemuan skema, pilih Mulai penemuan.
   - Untuk menghentikan penemuan skema, pilih Hapus penemuan.

Untuk memulai atau menghentikan penemuan skema pada bus acara menggunakan AWS CLI

- Untuk memulai penemuan skema, gunakan create-discoverer.

  Untuk menghentikan penemuan skema, gunakan delete-discoverer.

Menambahkan atau menghapus tag pada bus acara

Tag adalah label atribut kustom yang Anda atau AWS tetapkan ke AWS sumber daya. Gunakan tag untuk mengidentifikasi dan mengatur AWS sumber daya Anda. Untuk informasi selengkapnya, lihat EventBridge tag.

Untuk menambah atau menghapus tag dari bus acara menggunakan EventBridge konsol

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Bus peristiwa.
3. Pilih bus acara yang ingin Anda perbarui.
4. Pada halaman detail bus acara, pilih tab Tag, lalu pilih Kelola tag.
5. Lakukan salah satu dari cara berikut:
   - Untuk menambahkan tag:
     a. Pilih Tambahkan tanda baru.

b. Tentukan kunci dan nilai untuk tag
c. Pilih Perbarui.

- Untuk menghapus tag:

a. Untuk tag yang ingin Anda hapus, pilih Hapus.
b. Pilih Perbarui.

Untuk menambah atau menghapus tag dari bus acara menggunakan AWS CLI

- Untuk menambahkan tag, gunakan tag-resource.

  Untuk menghapus tag, gunakan untag-resource.

Menghapus bus EventBridge acara Amazon

Anda dapat menghapus bus acara khusus atau mitra. Anda tidak dapat menghapus bus acara default. Menghapus bus acara akan menghapus aturan yang terkait dengan bus acara tersebut.

Untuk menghapus bus acara menggunakan EventBridge konsol

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Bus peristiwa.
3. Pilih bus acara yang ingin Anda hapus.
4. Lakukan salah satu dari cara berikut:
   - Pilih Hapus.
   - Pilih nama bus acara.

     Pada halaman detail bus acara, pilih Hapus.

Izin untuk busEventBridge peristiwa Amazon

Bus peristiwa default di AWS akun Anda hanya mengizinkan peristiwa dari satu akun. Anda dapat memberikan izin tambahan untuk bus peristiwa dengan melampirkan kebijakan berbasis sumber daya. Dengan kebijakan berbasis sumber daya, Anda dapat mengizinkan `PutEvents`, `PutRule`, dan, `PutTargets` panggilan API dari akun lain. Anda juga dapat menggunakan kondisi IAM dalam kebijakan untuk memberikan izin kepada organisasi, menerapkan tag, atau filter peristiwa hanya

dari aturan atau akun tertentu. Anda dapat menetapkan kebijakan berbasis sumber daya untuk bus peristiwa ketika Anda membuat atau setelahnya.

EventBridgeAPI yang menerimaName parameter bus peristiwa sepertiPutRule,,PutTargetsDeleteRule,RemoveTargets,DisableRule, danEnableRule juga menerima ARN bus acara. Gunakan parameter ini untuk referensi bus peristiwa lintas akun atau lintas Wilayah melalui API. Misalnya, Anda dapat menghubungi PutRule untuk membuat aturan pada bus peristiwa di akun yang berbeda tanpa perlu menerima peran.

Anda dapat melampirkan contoh kebijakan dalam topik ini ke peran IAM untuk memberikan izin mengirim peristiwa ke akun atau Wilayah yang berbeda. Gunakan peran IAM untuk menetapkan kebijakan dan batasan kontrol organisasi tentang siapa yang dapat mengirim peristiwa dari akun Anda ke akun lain. Kami menyarankan untuk selalu menggunakan peran IAM ketika target aturan adalah bus acara. Anda dapat melampirkan peran IAM menggunakanPutTarget panggilan. Untuk informasi tentang cara membuat aturan untuk mengirim peristiwa ke akun atau Wilayah lain, lihat Mengirim dan menerima EventBridge acara Amazon antar AWS akun.

Topik

- Mengelola izin bus peristiwa
- Contoh kebijakan: Kirim peristiwa ke bus default di akun yang berbeda
- Contoh kebijakan: Kirim peristiwa ke bus kustomisasi di akun yang berbeda
- Contoh kebijakan: Kirim peristiwa ke bus peristiwa di akun yang sama
- Contoh kebijakan: Kirim peristiwa ke akun yang sama dan batasi pembaruan
- Contoh kebijakan: Kirim peristiwa hanya dari aturan khusus ke bus di Wilayah yang berbeda
- Contoh kebijakan: Kirim peristiwa hanya dari Wilayah tertentu ke Wilayah yang berbeda
- Contoh kebijakan: Tolak pengiriman peristiwa dari Daerah tertentu

Mengelola izin bus peristiwa

Gunakan prosedur berikut untuk memodifikasi izin bus peristiwa yang ada. Untuk informasi tentang cara menggunakanAWS CloudFormation untuk membuat kebijakan bus peristiwa, lihat AWS::Events::EventBusKebijakan.

Untuk mengelola izin bus peristiwa yang ada

1. Buka konsol Amazon EventBridge di https://console.aws.amazon.com/events/.

2. Di panel navigasi kiri, pilih Bus peristiwa.
3. Dalam Nama, pilih nama bus peristiwa untuk mengelola izinnya.

   Jika kebijakan sumber daya melekat pada bus peristiwa, maka kebijakan tersebut akan ditampilkan.
4. Pilih Mengelola izin, dan lakukan salah satu langkah berikut:
   - Masukkan kebijakan yang mencakup izin yang akan diberikan pada bus peristiwa. Anda dapat menyisipkan kebijakan dari sumber lain, atau memasukkan JSON untuk kebijakan.
   - Untuk menggunakan templat kebijakan, pilih Muat templat. Ubah kebijakan yang sesuai untuk lingkungan Anda, dan tambahkan tindakan tambahan yang mengizinkan utama dalam kebijakan untuk menggunakannya.
5. Pilih Perbarui.

Templat memberikan contoh pernyataan kebijakan yang dapat Anda sesuaikan untuk akun dan lingkungan Anda. Templat bukan kebijakan yang valid. Anda dapat memodifikasi templat untuk kasus penggunaan Anda, atau Anda dapat menyalin salah satu contoh kebijakan dan menyesuaikannya.

Templat memuat kebijakan yang mencakup contoh bagaimana cara memberikan izin pada akun untuk menggunakan `PutEvents` tindakan, cara memberikan izin pada organisasi, dan cara memberikan izin pada akun untuk mengelola aturan dalam akun itu sendiri. Anda dapat menyesuaikan templat untuk akun tertentu, kemudian menghapus bagian lain dari templat. Contoh kebijakan lainnya disertakan dalam topik ini.

Jika Anda mencoba untuk memperbarui izin bus namun terdapat kesalahan pada kebijakan, pesan yang muncul akan menunjukkan masalahnya.

```
 ### Choose which sections to include in the policy to match your use case. ###
 ### Be sure to remove all lines that start with ###, including the ### at the end of
the line. ###

 ### The policy must include the following: ###

 {
   "Version": "2012-10-17",
   "Statement": [

     ### To grant permissions for an account to use the PutEvents action, include the
following, otherwise delete this section: ###
```

```
    {

      "Sid": "AllowAccountToPutEvents",
      "Effect": "Allow",
      "Principal": {
        "AWS": "<ACCOUNT_ID>"
      },
      "Action": "events:PutEvents",
      "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/default"
    },

    ### Include the following section to grant permissions to all members of your AWS
Organizations to use the PutEvents action ###

    {
      "Sid": "AllowAllAccountsFromOrganizationToPutEvents",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "events:PutEvents",
      "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/default",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "o-yourOrgID"
        }
      }
    },

    ### Include the following section to grant permissions to the account to manage
the rules created in the account ###

    {
      "Sid": "AllowAccountToManageRulesTheyCreated",
      "Effect": "Allow",
      "Principal": {
        "AWS": "<ACCOUNT_ID>"
      },
      "Action": [
        "events:PutRule",
        "events:PutTargets",
        "events:DeleteRule",
        "events:RemoveTargets",
        "events:DisableRule",
        "events:EnableRule",
```

```
          "events:TagResource",
          "events:UntagResource",
          "events:DescribeRule",
          "events:ListTargetsByRule",
          "events:ListTagsForResource"],
        "Resource": "arn:aws:events:us-east-1:123456789012:rule/default",
        "Condition": {
          "StringEqualsIfExists": {
            "events:creatorAccount": "<ACCOUNT_ID>"
          }
        }
    }]
  }
```

Contoh kebijakan: Kirim peristiwa ke bus default di akun yang berbeda

Contoh kebijakan berikut ini memberikan izin akun 111122223333 untuk mempublikasikan peristiwa ke bus peristiwa default di akun 123456789012.

```
{
   "Version": "2012-10-17",
   "Statement": [
       {
        "Sid": "sid1",
        "Effect": "Allow",
        "Principal": {"AWS":"arn:aws:iam::111112222333:root"},
        "Action": "events:PutEvents",
        "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/default"
        }
    ]
  }
```

Contoh kebijakan: Kirim peristiwa ke bus kustomisasi di akun yang berbeda

Contoh kebijakan berikut memberikan izin pada akun 111122223333 untuk mempublikasikan peristiwa ke `central-event-bus` akun 123456789012, tetapi hanya untuk acara dengan nilai sumber yang ditetapkan ke `com.exampleCorp.webStore` dan `detail-type` diatur ke `newOrderCreated`.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Sid": "WebStoreCrossAccountPublish",
      "Effect": "Allow",
      "Action": [
        "events:PutEvents"
      ],
      "Principal": {
        "AWS": "arn:aws:iam::111112222333:root"
      },
      "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/central-event-bus",
      "Condition": {
        "StringEquals": {
          "events:detail-type": "newOrderCreated",
          "events:source": "com.exampleCorp.webStore"
        }
      }
    }
  ]
}
```

Contoh kebijakan: Kirim peristiwa ke bus peristiwa di akun yang sama

Contoh kebijakan berikut melekat pada bus peristiwa bernama`CustomBus1` memungkinkan bus peristiwa untuk menerima peristiwa dari akun dan Wilayah yang sama.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutEvents"
      ],
      "Resource": [
        "arn:aws:events:us-east-1:123456789:event-bus/CustomBus1"
      ]
    }
  ]
}
```

Contoh kebijakan: Kirim peristiwa ke akun yang sama dan batasi pembaruan

Contoh kebijakan berikut ini memberikan izin pada akun 123456789012 untuk membuat, menghapus, memperbarui, menonaktifkan dan mengaktifkan aturan, dan menambahkan atau menghapus target. Kebijakan tersebut membatasi aturan-aturan yang sesuai terhadap peristiwa dengan sumber `com.exampleCorp.webStore`, dan menggunakan `"events:creatorAccount": "${aws:PrincipalAccount}"` untuk memastikan bahwa hanya akun 123456789012 yang dapat memodifikasi aturan dan target ini setelah dibuat.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "InvoiceProcessingRuleCreation",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Action": [
        "events:PutRule",
        "events:DeleteRule",
        "events:DescribeRule",
        "events:DisableRule",
        "events:EnableRule",
        "events:PutTargets",
        "events:RemoveTargets"
      ],
      "Resource": "arn:aws:events:us-east-1:123456789012:rule/central-event-bus/*",
      "Condition": {
        "StringEqualsIfExists": {
          "events:creatorAccount": "${aws:PrincipalAccount}",
          "events:source": "com.exampleCorp.webStore"
        }
      }
    }
  ]
}
```

Contoh kebijakan: Kirim peristiwa hanya dari aturan khusus ke bus di Wilayah yang berbeda

Contoh kebijakan berikut mengizinkan akun 111122223333 untuk mengirim peristiwa yang sesuai dengan aturan bernama `SendToUSE1AnotherAccount` di Middle East (Bahrain) (Middle East (Bahrain)) dan US West (Oregon) untuk bus peristiwa bernama `CrossRegionBus` di US East (N. Virginia) pada akun 123456789012. Contoh kebijakan ditambahkan ke bus peristiwa bernama `CrossRegionBus` pada akun 123456789012. Kebijakan ini memperbolehkan peristiwa hanya jika sesuai dengan aturan yang ditentukan untuk bus peristiwa di akun 111122223333. Pernyataan `Condition` membatasi peristiwa hanya untuk peristiwa yang sesuai dengan aturan yang mengandung aturan khusus ARN.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowSpecificRulesAsCrossRegionSource",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111112222333:root"
      },
      "Action": "events:PutEvents",
      "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/CrossRegionBus",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": [
            "arn:aws:events:us-west-2:111112222333:rule/CrossRegionBus/
SendToUSE1AnotherAccount",
            "arn:aws:events:me-south-1:111112222333:rule/CrossRegionBus/
SendToUSE1AnotherAccount"
          ]
        }
      }
    }
  ]
}
```

Contoh kebijakan: Kirim peristiwa hanya dari Wilayah tertentu ke Wilayah yang berbeda

Contoh kebijakan berikut mengizinkan akun 111122223333 untuk mengirim semua peristiwa yang dihasilkan di Middle East (Bahrain) (Middle East (Bahrain)) dan US West (Oregon) untuk bus peristiwa bernama `CrossRegionBus` pada akun 123456789012 di US East (N. Virginia). Akun 111122223333 tidak memiliki izin untuk mengirim peristiwa yang dihasilkan di Wilayah lain.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCrossRegionEventsFromUSWest2AndMESouth1",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111112222333:root"
      },
      "Action": "events:PutEvents",
      "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/CrossRegionBus",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": [
            "arn:aws:events:us-west-2:*:*",
            "arn:aws:events:me-south-1:*:*"
          ]
        }
      }
    }
  ]
}
```

Contoh kebijakan: Tolak pengiriman peristiwa dari Daerah tertentu

Contoh kebijakan berikut melekat pada bus peristiwa bernama `CrossRegionBus` di akun 123456789012 yang mengizinkan bus peristiwa untuk menerima peristiwa dari akun 111122223333, namun tidak peristiwa yang dihasilkan di US West (Oregon).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "1AllowAnyEventsFromAccount111112222333",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111112222333:root"
      },
      "Action": "events:PutEvents",
      "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/CrossRegionBus"
    },
    {
      "Sid": "2DenyAllCrossRegionUSWest2Events",
      "Effect": "Deny",
      "Principal": {
        "AWS": "*"
      },
      "Action": "events:PutEvents",
      "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/CrossRegionBus",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": [
            "arn:aws:events:us-west-2:*:*"
          ]
        }
      }
    }
  ]
}
```

Buat AWS CloudFormation template dari bus EventBridge acara Amazon

AWS CloudFormationmemungkinkan Anda mengonfigurasi dan mengelola AWS sumber daya Anda di seluruh akun dan wilayah secara terpusat dan berulang dengan memperlakukan infrastruktur sebagai kode. CloudFormation melakukan ini dengan membiarkan Anda membuat template, yang menentukan sumber daya yang ingin Anda sediakan dan kelola.

EventBridge memungkinkan Anda untuk membuat template dari bus acara yang ada di akun Anda, sebagai bantuan untuk membantu Anda mulai mengembangkan CloudFormation template. Selain itu, EventBridge berikan opsi untuk memasukkan aturan yang terkait dengan bus acara itu di template Anda. Anda kemudian dapat menggunakan template ini sebagai dasar untuk membuat tumpukan sumber daya yang CloudFormation dikelola.

Untuk informasi lebih lanjut tentang CloudFormation lihat Panduan AWS CloudFormation Pengguna.

> **Note**
>
> EventBridge tidak menyertakan aturan terkelola dalam template yang dihasilkan.

Anda juga dapat membuat template dari satu atau beberapa aturan yang terdapat dalam bus acara yang dipilih.

Untuk menghasilkan CloudFormation template dari bus acara

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Bus peristiwa.
3. Pilih bus acara dari mana Anda ingin membuat CloudFormation template.
4. Dari menu Tindakan, pilih CloudFormation Template, lalu pilih format mana yang EventBridge ingin Anda buat template di: JSON atau YAMAL.

   EventBridge menampilkan template, yang dihasilkan dalam format yang dipilih. Secara default, semua aturan yang terkait dengan bus acara disertakan dalam template.

   - Untuk membuat template tanpa menyertakan aturan, batalkan pilihan Sertakan aturan pada ini EventBus.
5. EventBridge memberi Anda pilihan untuk mengunduh file template, atau menyalin template ke clipboard.

   - Untuk mengunduh file templat, pilih Unduh.
   - Untuk menyalin template ke clipboard, pilih Salin.
6. Untuk keluar dari template, pilih Batal.

Setelah Anda menyesuaikan AWS CloudFormation template Anda seperlunya untuk kasus penggunaan Anda, Anda dapat menggunakannya untuk membuat tumpukan. CloudFormation

Pertimbangan saat menggunakan CloudFormation template yang dihasilkan dari Amazon EventBridge

Pertimbangkan faktor-faktor berikut saat menggunakan CloudFormation template yang Anda buat dari bus acara:

- EventBridge tidak menyertakan kata sandi apa pun dalam template generate.

  Anda dapat mengedit templat untuk menyertakan parameter templat yang memungkinkan pengguna menentukan kata sandi atau informasi sensitif lainnya saat menggunakan templat untuk membuat atau memperbarui CloudFormation tumpukan.

  Selain itu, pengguna dapat menggunakan Secrets Manager untuk membuat rahasia di wilayah yang diinginkan dan kemudian mengedit template yang dihasilkan untuk menggunakan parameter dinamis.
- Target dalam template yang dihasilkan tetap persis seperti yang ditentukan dalam bus acara asli. Hal ini dapat menyebabkan masalah lintas wilayah jika Anda tidak mengedit template dengan tepat sebelum menggunakannya untuk membuat tumpukan di wilayah lain.

  Selain itu, template yang dihasilkan tidak akan membuat target hilir secara otomatis.

EventBridge Acara Amazon

Peristiwa menunjukkan perubahan dalam lingkungan seperti lingkungan AWS , layanan atau aplikasi mitra SaaS, atau salah satu aplikasi atau layanan Anda. Berikut ini adalah contoh peristiwa:

- Amazon EC2 menghasilkan peristiwa ketika keadaan instans perubahan dari tertunda menjadi dijalankan.
- Amazon EC2 Auto Scaling menghasilkan peristiwa ketika meluncurkan atau mengakhiri instans.
- AWS CloudTrail menerbitkan peristiwa saat Anda melakukan panggilan API.

Anda juga dapat mengatur peristiwa terjadwal yang dihasilkan secara berkala.

Untuk daftar layanan yang menghasilkan peristiwa, termasuk contoh peristiwa dari setiap layanan, lihat Acara dari AWS layanan dan ikuti tautan dalam tabel.

Peristiwa direpresentasikan sebagai objek JSON dan mereka semua memiliki struktur yang sama, dan bidang tingkat atas yang sama.

Konten dari bidang tingkat atas detail berbeda bergantung pada layanan yang menghasilkan peristiwa dan apa peristiwanya. Kombinasi dari sumber dan bidang jenis detail berfungsi untuk mengidentifikasi bidang dan nilai yang ditemukan dalam bidang detail. Untuk contoh peristiwa yang dihasilkan oleh AWS layanan, lihatAcara dari AWS layanan.

Topik

- Referensi struktur acara
- Menambahkan EventBridge acara Amazon dengan PutEvents
- Acara dari AWS layanan
- Menerima acara dari mitra SaaS dengan Amazon EventBridge
- Mendebug pengiriman EventBridge acara Amazon

Video berikut menjelaskan dasar-dasar acara: Apa itu acara

Video berikut mencakup cara acara mencapai EventBridge: Dari mana datangnya peristiwa

Referensi struktur acara

Bidang berikut ini muncul dalam peristiwa:

```
{
  "version": "0",
  "id": "UUID",
  "detail-type": "event name",
  "source": "event source",
  "account": "ARN",
  "time": "timestamp",
  "region": "region",
  "resources": [
    "ARN"
  ],
  "detail": {
    JSON object
  }
}
```

versi

Secara default, ini diatur ke 0 (nol) di semua peristiwa.

id

Versi 4 UUID yang dihasilkan untuk setiap peristiwa. Anda dapat menggunakan `id` untuk melacak peristiwa saat mereka bergerak melalui aturan ke target.

jenis-detail:

Mengidentifikasi, dalam kombinasi dengan bidang sumber, bidang dan nilai yang muncul di bidang detail.

Acara yang disampaikan oleh CloudTrail memiliki `AWS API Call via CloudTrail` nilai untuk`detail-type`.

sumber

Mengidentifikasi layanan yang menghasilkan peristiwa. Semua peristiwa yang datang dari layanan AWS dimulai dengan “aws.” Peristiwa yang dihasilkan pelanggan dapat memiliki nilai berapa pun di sini, selama tidak dimulai dengan “aws.” Kami merekomendasikan penggunaan string nama domain terbalik gaya nama paket Java.

Untuk menemukan nilai yang benar `source` untuk AWS layanan, lihat tabel tombol kondisi, pilih layanan dari daftar, dan cari awalan layanan. Misalnya, `source` nilai untuk Amazon CloudFront adalah`aws.cloudfront`.

akun

Nomor 12 digit yang mengidentifikasi AWS akun.

Waktu

Peristiwa timestamp, yang dapat ditentukan oleh layanan yang berasal dari peristiwa. Jika peristiwa mencakup interval waktu, layanan dapat melaporkan waktu mulai, sehingga nilai ini mungkin sebelum waktu peristiwa diterima.

region

Mengidentifikasi AWS Wilayah tempat acara tersebut berasal.

sumber daya

Array JSON yang berisi ARN yang mengidentifikasi sumber daya yang terlibat dalam peristiwa tersebut. Layanan yang menghasilkan peristiwa menentukan apakah akan memasukkan ARN tersebut. Sebagai contoh, perubahan keadaan instans Amazon EC2 termasuk ARN instans Amazon EC2, peristiwa Auto Scaling termasuk ARN untuk kedua instans dan kelompok Auto Scaling, tapi panggilan API dengan AWS CloudTrail tidak mencakup ARN sumber daya.

detail

Objek JSON yang berisi informasi tentang peristiwa. Layanan yang menghasilkan acara menentukan konten bidang ini. Konten detail bisa sesederhana dua bidang. AWS Peristiwa panggilan API memiliki objek detail dengan sekitar 50 bidang bersarang beberapa level.

Example Contoh: Pemberitahuan perubahan status instans Amazon EC2

Peristiwa berikut di Amazon EventBridge menunjukkan instans Amazon EC2 dihentikan.

```
{
  "version": "0",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "EC2 Instance State-change Notification",
  "source": "aws.ec2",
  "account": "111122223333",
  "time": "2017-12-22T18:43:48Z",
```

```
  "region": "us-west-1",
  "resources": [
    "arn:aws:ec2:us-west-1:123456789012:instance/i-1234567890abcdef0"
  ],
  "detail": {
    "instance-id": " i-1234567890abcdef0",
    "state": "terminated"
  }
}
```

Informasi minimum yang diperlukan untuk acara kustom yang valid

Saat Anda membuat acara khusus, mereka harus menyertakan bidang berikut:

```
{
  "detail-type": "event name",
  "source": "event source",
  "detail": {
  }
}
```

- `detail`— Objek JSON yang berisi informasi tentang acara tersebut. Itu bisa`"{}"`.

> **Note**
>
> PutEventsmenerima data dalam format JSON. Untuk tipe data nomor JSON (integer), kendala adalah: nilai minimum -9.223.372.036.854.775.808 dan nilai maksimum 9.223.372.036.854.775.807.

- `detail-type`— String yang mengidentifikasi jenis acara.
- `source`— String yang mengidentifikasi sumber acara. Peristiwa yang dihasilkan pelanggan dapat memiliki nilai berapa pun di sini, selama tidak dimulai dengan “aws.” Kami merekomendasikan penggunaan string nama domain terbalik gaya nama paket Java.

Menambahkan EventBridge acara Amazon dengan **`PutEvents`**

`PutEvents`Tindakan mengirimkan beberapa peristiwa ke EventBridge dalam satu permintaan. Untuk informasi selengkapnya, lihat PutEventsReferensi Amazon EventBridge API dan put-events di Referensi AWS CLI Perintah.

Setiap permintaan `PutEvents` dapat mendukung sejumlah entri yang terbatas. Untuk informasi selengkapnya, lihat EventBridge Kuota Amazon. Operasi `PutEvents` mencoba untuk memproses semua entri dalam urutan alami permintaan. Setelah Anda meneleponPutEvents, EventBridge berikan setiap peristiwa ID unik.

Topik

- Menangani kegagalan dengan PutEvents
- Mengirim acara menggunakan AWS CLI
- Menghitung ukuran entri EventBridge PutEvents acara Amazon

Contoh kode Java berikut mengirimkan dua peristiwa identik ke EventBridge.

AWS SDK for Java Version 2.x

```
EventBridgeClient eventBridgeClient =
    EventBridgeClient.builder().build();

PutEventsRequestEntry requestEntry = PutEventsRequestEntry.builder()
    .resources("resource1", "resource2")
    .source("com.mycompany.myapp")
    .detailType("myDetailType")
    .detail("{ \"key1\": \"value1\", \"key2\": \"value2\" }")
    .build();

List <
PutEventsRequestEntry > requestEntries = new ArrayList <
PutEventsRequestEntry > ();
requestEntries.add(requestEntry);

PutEventsRequest eventsRequest = PutEventsRequest.builder()
    .entries(requestEntries)
    .build();

PutEventsResponse result = eventBridgeClient.putEvents(eventsRequest);

for (PutEventsResultEntry resultEntry: result.entries()) {
    if (resultEntry.eventId() != null) {
        System.out.println("Event Id: " + resultEntry.eventId());
    } else {
        System.out.println("PutEvents failed with Error Code: " +
 resultEntry.errorCode());
```

```
    }
}
```

AWS SDK for Java Version 1.0

```
EventBridgeClient eventBridgeClient =
    EventBridgeClient.builder().build();

PutEventsRequestEntry requestEntry = new PutEventsRequestEntry()
        .withTime(new Date())
        .withSource("com.mycompany.myapp")
        .withDetailType("myDetailType")
        .withResources("resource1", "resource2")
        .withDetail("{ \"key1\": \"value1\", \"key2\": \"value2\" }");

PutEventsRequest request = new PutEventsRequest()
        .withEntries(requestEntry, requestEntry);

PutEventsResult result = awsEventsClient.putEvents(request);

for (PutEventsResultEntry resultEntry : result.getEntries()) {
    if (resultEntry.getEventId() != null) {
        System.out.println("Event Id: " + resultEntry.getEventId());
    } else {
        System.out.println("Injection failed with Error Code: " +
 resultEntry.getErrorCode());
    }
}
```

Setelah Anda menjalankan kode ini, hasil `PutEvents` termasuk array entri respons. Setiap entri dalam array respon sesuai dengan entri dalam array permintaan dalam urutan dari awal sampai akhir permintaan dan respons. Array `Entries` respons selalu mencakup jumlah entri yang sama sebagai array permintaan.

Menangani kegagalan dengan **`PutEvents`**

Secara default, jika entri individu dalam permintaan gagal, EventBridge terus memproses sisa entri dalam permintaan. Array `Entries` respons dapat mencakup entri yang berhasil dan yang tidak berhasil. Anda harus mendeteksi entri yang tidak berhasil dan menyertakannya ke dalam panggilan berikutnya.

Entri hasil yang berhasil mencakup nilai `Id`, dan entri hasil yang tidak berhasil mencakup nilai `ErrorCode` dan `ErrorMessage`. `ErrorCode` menguraikan jenis kesalahan.`ErrorMessage` menyediakan informasi lebih lanjut tentang kesalahan. Contoh berikut ini memiliki tiga entri hasil untuk permintaan `PutEvents`. Entri kedua tidak berhasil.

```
{
    "FailedEntryCount": 1,
    "Entries": [
        {
            "EventId": "11710aed-b79e-4468-a20b-bb3c0c3b4860"
        },
        {   "ErrorCode": "InternalFailure",
            "ErrorMessage": "Internal Service Failure"
        },
        {
            "EventId": "d804d26a-88db-4b66-9eaf-9a11c708ae82"
        }
    ]
}
```

> **Note**
>
> Jika Anda menggunakan `PutEvents` untuk mempublikasikan acara ke bus acara yang tidak ada, pencocokan EventBridge acara tidak akan menemukan aturan yang sesuai dan akan membatalkan acara tersebut. Meskipun EventBridge akan mengirim `200` respons, itu tidak akan gagal permintaan atau menyertakan acara dalam `FailedEntryCount` nilai respons permintaan.

Anda dapat menyertakan entri yang tidak berhasil dalam permintaan `PutEvents` berikutnya. Pertama, untuk mengetahui apakah ada entri gagal dalam permintaan, periksa parameter `FailedRecordCount` dalam `PutEventsResult`. Jika bukan nol, maka Anda dapat menambahkan masing-masing `Entry`yang memiliki `ErrorCode` yang bukan null untuk permintaan berikutnya. Contoh berikut ini menunjukkan penangan kegagalan.

```
PutEventsRequestEntry requestEntry = new PutEventsRequestEntry()
        .withTime(new Date())
        .withSource("com.mycompany.myapp")
        .withDetailType("myDetailType")
        .withResources("resource1", "resource2")
```

```
        .withDetail("{ \"key1\": \"value1\", \"key2\": \"value2\" }");

List<PutEventsRequestEntry> putEventsRequestEntryList = new ArrayList<>();
for (int i = 0; i < 3; i++) {
    putEventsRequestEntryList.add(requestEntry);
}

PutEventsRequest putEventsRequest = new PutEventsRequest();
putEventsRequest.withEntries(putEventsRequestEntryList);
PutEventsResult putEventsResult = awsEventsClient.putEvents(putEventsRequest);

while (putEventsResult.getFailedEntryCount() > 0) {
    final List<PutEventsRequestEntry> failedEntriesList = new ArrayList<>();
    final List<PutEventsResultEntry> PutEventsResultEntryList =
 putEventsResult.getEntries();
    for (int i = 0; i < PutEventsResultEntryList.size(); i++) {
        final PutEventsRequestEntry putEventsRequestEntry =
 putEventsRequestEntryList.get(i);
        final PutEventsResultEntry putEventsResultEntry =
 PutEventsResultEntryList.get(i);
        if (putEventsResultEntry.getErrorCode() != null) {
            failedEntriesList.add(putEventsRequestEntry);
        }
    }
    putEventsRequestEntryList = failedEntriesList;
    putEventsRequest.setEntries(putEventsRequestEntryList);
    putEventsResult = awsEventsClient.putEvents(putEventsRequest);
    }
```

Mengirim acara menggunakan AWS CLI

Anda dapat menggunakan AWS CLI untuk mengirim acara khusus EventBridge agar dapat diproses. Contoh berikut menempatkan satu acara khusus ke dalam EventBridge:

```
aws events put-events \
--entries '[{"Time": "2016-01-14T01:02:03Z", "Source": "com.mycompany.myapp",
 "Resources": ["resource1", "resource2"], "DetailType": "myDetailType", "Detail":
 "{ \"key1\": \"value1\", \"key2\": \"value2\" }"}]'
```

Anda juga dapat membuat file JSON yang berisi peristiwa kustom.

```
[
```

```
  {
    "Time": "2016-01-14T01:02:03Z",
    "Source": "com.mycompany.myapp",
    "Resources": [
      "resource1",
      "resource2"
    ],
    "DetailType": "myDetailType",
    "Detail": "{ \"key1\": \"value1\", \"key2\": \"value2\" }"
  }
]
```

Kemudian, untuk menggunakan AWS CLI untuk membaca entri dari file ini dan mengirim acara, pada prompt perintah, ketik:

```
aws events put-events --entries file://entries.json
```

Menghitung ukuran entri EventBridge PutEvents acara Amazon

Anda dapat mengirim acara khusus EventBridge dengan menggunakan `PutEvents` tindakan. Anda dapat membuat batch beberapa entri peristiwa menjadi satu permintaan untuk efisiensi. Ukuran entri total harus kurang dari 256KB. Anda dapat menghitung ukuran entri sebelum Anda mengirim peristiwa.

> **Note**
>
> Batas ukuran dikenakan pada entri. Bahkan jika entri kurang dari batas ukuran, acara di selalu EventBridge lebih besar dari ukuran entri karena karakter dan kunci yang diperlukan dari representasi JSON dari acara tersebut. Untuk informasi selengkapnya, lihat EventBridge Acara Amazon.

EventBridge menghitung `PutEventsRequestEntry` ukuran sebagai berikut:

- Jika ditentukan, parameter `Time` adalah 14 byte.
- Parameter `Source` dan `DetailType` adalah jumlah byte untuk bentuk terenkodekan UTF-8 mereka.
- Jika ditentukan, parameter `Detail` adalah jumlah byte untuk bentuk terenkodekan UTF-8-nya.
- Jika ditentukan, setiap entri parameter `Resources` adalah jumlah byte untuk bentuk terenkodekan UTF-8-nya.

Contoh kode Java berikut ini menghitung ukuran objek `PutEventsRequestEntry` yang diberikan.

```
int getSize(PutEventsRequestEntry entry) {
    int size = 0;
    if (entry.getTime() != null) {
        size += 14;
    }
    size += entry.getSource().getBytes(StandardCharsets.UTF_8).length;
    size += entry.getDetailType().getBytes(StandardCharsets.UTF_8).length;
    if (entry.getDetail() != null) {
        size += entry.getDetail().getBytes(StandardCharsets.UTF_8).length;
    }
    if (entry.getResources() != null) {
        for (String resource : entry.getResources()) {
```

```
            if (resource != null) {
                size += resource.getBytes(StandardCharsets.UTF_8).length;
            }
        }
    }
    return size;
}
```

> **Note**
>
> Jika ukuran entri lebih besar dari 256KB, kami sarankan untuk mengunggah acara ke bucket Amazon S3 dan menyertakan entri di entri. `Object URL PutEvents`

Acara dari AWS layanan

Banyak AWS layanan menghasilkan acara yang EventBridge menerima. Ketika AWS layanan di akun Anda memancarkan suatu peristiwa, layanan tersebut masuk ke bus acara default akun Anda.

Pengiriman acara dari AWS layanan

Setiap AWS layanan yang menghasilkan acara mengirimkannya EventBridge sebagai upaya terbaik atau upaya pengiriman yang tahan lama.

- Penyampaian upaya terbaik berarti bahwa layanan mencoba mengirim semua acara ke EventBridge, tetapi dalam beberapa kasus yang jarang terjadi suatu peristiwa mungkin tidak disampaikan.
- Pengiriman yang tahan lama berarti layanan akan berhasil mengantarkan acara EventBridge setidaknya sekali.

  EventBridge akan menerima semua acara yang valid dalam kondisi normal. Dalam kasus di mana acara tidak dapat disampaikan karena gangguan EventBridge layanan, mereka akan dicoba lagi nanti oleh AWS layanan hingga 24 jam.

Setelah acara dikirimkan EventBridge, EventBridge cocokkan dengan aturan dan kemudian ikuti kebijakan coba lagi dan antrean huruf mati yang ditentukan untuk target acara.

Untuk daftar AWS layanan yang menghasilkan peristiwa, lihat???.

Mengakses acara AWS layanan melalui AWS CloudTrail

AWS CloudTrail adalah layanan yang secara otomatis merekam peristiwa seperti panggilan AWS API. Anda dapat membuat EventBridge aturan yang menggunakan informasi dari CloudTrail. Untuk informasi lebih lanjut tentang CloudTrail, lihat Apa itu AWS CloudTrail? .

Semua acara yang disampaikan oleh CloudTrail memiliki `AWS API Call via CloudTrail` nilai untuk`detail-type`.

Untuk merekam peristiwa dengan `detail-type` nilai`AWS API Call via CloudTrail`, CloudTrail jejak dengan logging diaktifkan diperlukan.

Saat menggunakan CloudTrail Amazon S3, Anda perlu mengonfigurasi CloudTrail untuk mencatat peristiwa data. Untuk informasi selengkapnya, lihat Mengaktifkan pencatatan CloudTrail peristiwa untuk bucket dan objek S3.

Beberapa kejadian dalam AWS layanan dapat dilaporkan EventBridge baik oleh layanan itu sendiri maupun oleh CloudTrail. Misalnya, panggilan API Amazon EC2 yang memulai atau menghentikan instance menghasilkan EventBridge peristiwa serta peristiwa melalui. CloudTrail

CloudTrail mendukung penelepon API dan pemilik sumber daya untuk menerima peristiwa di bucket Amazon S3 mereka dengan membuat jejak, dan mengirimkan peristiwa ke penelepon API melalui. EventBridge Pemilik sumber daya selain penelepon API dapat memantau panggilan API lintas akun. EventBridge CloudTrailIntegrasi dengan EventBridge menyediakan cara mudah untuk mengatur alur kerja berbasis aturan otomatis dalam menanggapi peristiwa.

Anda tidak dapat menggunakan AWS peristiwa panggilan API Put*Events yang berukuran lebih besar dari 256 KB sebagai pola peristiwa karena ukuran maksimum permintaan Put*Events adalah 256 KB. Untuk informasi selengkapnya tentang panggilan API yang dapat Anda gunakan, lihat layanan dan integrasi yang CloudTrail didukung.

Menerima acara manajemen hanya-baca dari layanan AWS

Anda dapat mengatur aturan pada bus acara default atau kustom Anda untuk menerima acara manajemen hanya-baca dari AWS layanan melalui. CloudTrail Acara manajemen memberikan visibilitas ke dalam operasi manajemen yang dilakukan pada sumber daya di AWS akun Anda. Ini juga dikenal sebagai operasi pesawat kontrol. Untuk informasi selengkapnya, lihat Pencatatan peristiwa manajemen di Panduan CloudTrail Pengguna.

Untuk setiap aturan pada bus acara default atau kustom, Anda dapat mengatur status aturan untuk mengontrol jenis peristiwa yang akan diterima:

- Nonaktifkan aturan sehingga EventBridge tidak cocok dengan acara terhadap aturan.
- Aktifkan aturan agar EventBridge sesuai dengan peristiwa dengan aturan, kecuali untuk acara AWS manajemen hanya-baca yang disampaikan. CloudTrail
- Aktifkan aturan agar EventBridge sesuai dengan semua peristiwa dengan aturan, termasuk peristiwa manajemen hanya-baca yang disampaikan. CloudTrail

Bus acara mitra tidak menerima AWS acara.

Beberapa hal yang perlu dipertimbangkan ketika memutuskan apakah akan menerima acara manajemen hanya-baca:

- Peristiwa manajemen hanya-baca tertentu, seperti AWS Key Management Service `GetKeyPolicy` dan`DescribeKey`, atau IAM `GetPolicy` dan `GetRole` peristiwa, terjadi pada volume yang jauh lebih tinggi daripada peristiwa perubahan biasa.
- Anda mungkin sudah menerima acara manajemen hanya-baca, jika acara tersebut tidak dimulai dengan`Describe`,`Get`, atau. `List` Misalnya, peristiwa dari AWS STS API berikut adalah peristiwa perubahan, bahkan dianggap dimulai dengan kata kerja`Get`:
  - `GetFederationToken`
  - `GetSessionToken`

  Untuk daftar peristiwa manajemen hanya-baca yang tidak mematuhi,, atau konvensi `List` penamaan `DescribeGet`, berdasarkan AWS layanan, lihat. ???

Untuk membuat aturan yang menerima peristiwa manajemen hanya-baca menggunakan CLI AWS

- Gunakan `put-rule` perintah untuk membuat atau memperbarui aturan, menggunakan parameter untuk:
  - Tentukan bahwa aturan tersebut termasuk dalam bus acara default, atau bus acara khusus tertentu
  - Tetapkan status aturan sebagai `ENABLED_WITH_ALL_CLOUDTRAIL_MANAGEMENT_EVENTS`

  ```
  aws events put-rule --name "ruleForManagementEvents" --event-bus-name
  "default" --state "ENABLED_WITH_ALL_CLOUDTRAIL_MANAGEMENT_EVENTS"
  ```

> **Note**
>
> Mengaktifkan aturan untuk acara CloudWatch manajemen didukung melalui AWS CLI AWS CloudFormation dan template saja.

Example

Contoh berikut menggambarkan bagaimana mencocokkan dengan peristiwa tertentu. Praktik terbaik adalah mendefinisikan aturan khusus untuk mencocokkan acara tertentu, untuk kejelasan dan kemudahan pengeditan.

Dalam hal ini, aturan khusus cocok dengan acara `AssumeRole` manajemen dari AWS Security Token Service.

```
{
    "source" : [ "aws.sts" ],
    "detail-type": ["AWS API Call via CloudTrail"],
    "detail" : {
        "eventName" : ["AssumeRole"]
    }
}
```

AWS layanan yang menghasilkan peristiwa

Tabel berikut menunjukkan AWS layanan yang menghasilkan peristiwa. Pilih nama layanan untuk melihat informasi selengkapnya tentang cara layanan tersebut dan EventBridge bekerja sama.

Setiap AWS layanan yang menghasilkan acara mengirimkannya EventBridge sebagai upaya terbaik atau upaya pengiriman yang tahan lama. Untuk informasi selengkapnya, lihat ???.

Tabel ini mencakup representasi AWS layanan yang mengirim acara ke EventBridge, tetapi tidak mencakup setiap layanan. Untuk layanan yang tidak terdaftar yang mengirim acara ke EventBridge, asumsikan pengiriman upaya terbaik.

Layanan	Jenis Percobaan
Alexa for Business	Upaya terbaik
AWS Account Management	Upaya terbaik

Layanan	Jenis Percobaan
Amazon API Gateway	Upaya terbaik
AWS AppConfig	Upaya terbaik
Amazon AppFlow	Upaya terbaik
Application Auto Scaling	Upaya terbaik
AWS Profiler Biaya Aplikasi	Upaya terbaik
AWS Application Migration Service	Upaya terbaik
Amazon Athena	Upaya terbaik
AWS Backup	Upaya terbaik
AWS Batch	Tahan lama
Amazon Braket	Tahan lama
AWS Certificate Manager	Upaya terbaik
Amazon Chime	Upaya terbaik
Direktori Cloud Amazon	Upaya terbaik
AWS CloudFormation	Tahan lama
Amazon CloudFront	Upaya terbaik
AWS CloudHSM	Upaya terbaik
Amazon CloudSearch	Upaya terbaik
AWS CloudShell	Upaya terbaik
Acara dari AWS CloudTrail	Upaya terbaik
Amazon CloudWatch	Tahan lama

Layanan	Jenis Percobaan
Wawasan CloudWatch Aplikasi Amazon	Upaya terbaik
Monitor CloudWatch Internet Amazon	Upaya terbaik
CloudWatch Log Amazon	Upaya terbaik
Amazon CloudWatch Synthetics	Upaya terbaik
AWS CodeArtifact	Tahan lama
AWS CodeBuild	Upaya terbaik
AWS CodeCommit	Upaya terbaik
AWS CodeDeploy	Upaya terbaik
Amazon CodeGuru Profiler	Upaya terbaik
AWS CodePipeline	Upaya terbaik
AWS CodeStar	Upaya terbaik
CodeConnections	Upaya terbaik
Identitas Amazon Cognito	Upaya terbaik
Kolam pengguna Amazon Cognito	Upaya terbaik
Amazon Cognito Sync	Upaya terbaik
AWS Config	Upaya terbaik
Amazon Connect	Upaya terbaik
ID Suara Amazon Connect	Upaya terbaik
AWS Control Tower	Upaya terbaik
AWS Database Migration Service	Upaya terbaik

Layanan	Jenis Percobaan
AWS Data Exchange	Upaya terbaik
Amazon Data Lifecycle Manager	Upaya terbaik
AWS Data Pipeline	Upaya terbaik
AWS DataSync	Upaya terbaik
AWS Device Farm	Upaya terbaik
DevOpsGuru Amazon	Upaya terbaik
AWS Direct Connect	Upaya terbaik
AWS Directory Service	Upaya terbaik
Amazon DynamoDB	Upaya terbaik
AWS Elastic Beanstalk	Upaya terbaik
Toko Blok Elastis Amazon	Upaya terbaik
Modifikasi volume Amazon Elastic Block Store	Upaya terbaik
Amazon ElastiCache	Upaya terbaik
Amazon Elastic Compute Cloud (Amazon EC2)	Upaya terbaik
Amazon EC2 Auto Scaling	Upaya terbaik
Amazon EC2 Fleet	Upaya terbaik
Gangguan Instans Spot Amazon EC2	Upaya terbaik
Amazon Elastic Container Registry	Upaya terbaik
Layanan Kontainer Elastis Amazon	Tahan lama
AWS Elastic Disaster Recovery	Upaya terbaik

Layanan	Jenis Percobaan
Amazon Elastic File System	Upaya terbaik
Amazon Elastic Kubernetes Service	Upaya terbaik
Penyeimbang Beban Elastis	Upaya terbaik
Amazon Elastis MapReduce	Upaya terbaik
Amazon Elastic Transcoder	Upaya terbaik
AWS Elemental MediaConnect	Upaya terbaik
AWS Elemental MediaConvert	Tahan lama
AWS Elemental MediaLive	Upaya terbaik
AWS Elemental MediaPackage	Upaya terbaik
AWS Elemental MediaStore	Tahan lama
Amazon EMR	Upaya terbaik
Amazon EMR di EKS	Upaya terbaik
Amazon EMR Tanpa Server	Upaya terbaik
Aturan EventBridge terjadwal Amazon	Tahan lama
EventBridge Skema Amazon	Upaya terbaik
AWS Fault Injection Service	Upaya terbaik
Forecast	Upaya terbaik
Amazon GameLift	Upaya terbaik
AWS Glue	Upaya terbaik
AWS Glue DataBrew	Upaya terbaik

Layanan	Jenis Percobaan
AWS Ground Station	Upaya terbaik
Amazon GuardDuty	Upaya terbaik
AWS Health	Tahan lama
AWS HealthLake	Tahan lama
AWS Identity and Access Management (IAM)	Upaya terbaik
Penganalisis Akses IAM	Upaya terbaik
Amazon Inspector Klasik	Upaya terbaik
Amazon Inspector	Upaya terbaik
AWS IoT	Upaya terbaik
AWS IoT Analytics	Tahan lama
AWS IoT Greengrass V1	Upaya terbaik
AWS IoT Greengrass V2	Upaya terbaik
Layanan Video Interaktif Amazon	Upaya terbaik
Amazon Kinesis	Upaya terbaik
Amazon Data Firehose	Upaya terbaik
AWS Key Management Service Penghapusan CMK	Tahan lama
AWS Key Management Service Rotasi CMK	Upaya terbaik
AWS Key Management Service kedaluwarsa bahan kunci yang diimpor	Upaya terbaik
AWS Lambda	Upaya terbaik

Layanan	Jenis Percobaan
Amazon Location Service	Tahan lama
Amazon Machine Learning	Upaya terbaik
Amazon Macie	Upaya terbaik
Amazon Managed Blockchain	Upaya terbaik
AWS Managed Services	Upaya terbaik
AWS Management Console Masuk	Upaya terbaik
AWS Metering Marketplace	Upaya terbaik
AWS Migration Hub	Upaya terbaik
AWS Migration Hub Refactor Spaces	Upaya terbaik
AWS Pemantauan	Upaya terbaik
AWS Network Manager	Upaya terbaik
OpenSearch Layanan Amazon	Upaya terbaik
AWS OpsWorks	Tahan lama
AWS OpsWorks CM	Upaya terbaik
AWS Organizations	Upaya terbaik
Amazon Polly	Upaya terbaik
AWS Private Certificate Authority	Upaya terbaik
AWS Proton	Upaya terbaik
Amazon QLDB	Tahan lama
Amazon QuickSight	Upaya terbaik

Layanan	Jenis Percobaan
Amazon RDS	Upaya terbaik
AWS Tempat Sampah Daur Ulang	Upaya terbaik
Amazon Redshift	Tahan lama
API Data Amazon Redshift	Upaya terbaik
Amazon Redshift Tanpa Server	Upaya terbaik
AWS Resource Access Manager	Upaya terbaik
AWS Resource Groups	Upaya terbaik
AWS Resource Groups Tagging API	Upaya terbaik
Amazon Route 53	Upaya terbaik
Kesiapan Pemulihan Amazon Route 53	Upaya terbaik
Amazon SageMaker	Upaya terbaik
Savings Plans	Upaya terbaik
AWS Secrets Manager	Upaya terbaik
AWS Security Hub	Tahan lama
AWS Security Token Service	Upaya terbaik
AWS Server Migration Service	Upaya terbaik
AWS Service Catalog	Upaya terbaik
AWS Signer	Tahan lama
Layanan Email Sederhana Amazon	Upaya terbaik
Amazon Simple Storage Service (Amazon S3)	Tahan lama

Layanan	Jenis Percobaan
Amazon S3 Glacier	Upaya terbaik
Amazon S3 on Outposts	Upaya terbaik
Amazon Simple Queue Service	Upaya terbaik
Amazon Simple Notification Service	Upaya terbaik
Layanan Alur Kerja Sederhana Amazon	Upaya terbaik
AWS Step Functions	Upaya terbaik
AWS Storage Gateway	Tahan lama
AWS Support	Upaya terbaik
AWS Systems Manager	Upaya terbaik
Amazon Transcribe	Upaya terbaik
AWS Transfer Family	Upaya terbaik
AWS Transit Gateway	Upaya terbaik
Amazon Translate	Tahan lama
AWS Trusted Advisor	Upaya terbaik
AWS WAF	Upaya terbaik
AWS WAF Regional	Upaya terbaik
AWS Well-Architected Tool	Upaya terbaik
Amazon WorkDocs	Upaya terbaik
Amazon WorkSpaces	Upaya terbaik
AWS X-Ray	Upaya terbaik

Acara manajemen yang dihasilkan oleh AWS layanan

Secara umum, API yang menghasilkan peristiwa manajemen (atau hanya-baca) dimulai dengan kata kerja`Describe`,, `Get` atau. `List` Tabel di bawah ini mencantumkan AWS layanan dan peristiwa manajemen yang mereka hasilkan yang tidak mengikuti konvensi penamaan ini. Untuk informasi selengkapnya tentang acara manajemen, lihat???.

Acara manajemen yang tidak dimulai dengan**`Describe`**,**`Get`**, atau **`List`**

AWS Layanan daftar tabel berikut dan peristiwa manajemen yang mereka hasilkan yang tidak mengikuti konvensi penamaan khas dimulai dengan`Describe`,`Get`, atau`List`.

Layanan	Nama peristiwa	Jenis peristiwa
Alexa for Business	ResolveRoom	Panggilan API
Alexa for Business	SearchAddressBooks	Panggilan API
Alexa for Business	SearchContacts	Panggilan API
Alexa for Business	SearchDevices	Panggilan API
Alexa for Business	SearchProfiles	Panggilan API
Alexa for Business	SearchRooms	Panggilan API
Alexa for Business	SearchSkillGroups	Panggilan API
Alexa for Business	SearchUsers	Panggilan API
IAM Access Analyzer	ValidatePolicy	Panggilan API
AWS AdSpace Kamar Bersih	BatchGetSchema	Panggilan API
AWS Amplify Pembuat UI	ExportComponents	Panggilan API
AWS Amplify Pembuat UI	ExportForms	Panggilan API
AWS Amplify Pembuat UI	ExportThemes	Panggilan API
OpenSearch Layanan Amazon	BatchGetCollection	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
Amazon API Gateway	ExportApi	Panggilan API
AWS AppConfig	ValidateConfiguration	Panggilan API
Amazon AppFlow	RetrieveConnectorData	Panggilan API
Wawasan CloudWatch Aplikasi Amazon	UpdateApplicationDashboardConfiguration	Panggilan API
Amazon Athena	BatchGetNamedQuery	Panggilan API
Amazon Athena	BatchGetPreparedStatement	Panggilan API
Amazon Athena	BatchGetQueryExecution	Panggilan API
Amazon Athena	CheckQueryCompatibility	Panggilan API
Amazon Athena	ExportNotebook	Panggilan API
AWS Auto Scaling	AreScalableTargetsRegistered	Panggilan API
AWS Auto Scaling	Uji	Panggilan API
AWS Marketplace	SearchAgreements	Panggilan API
AWS Backup	CreateLegalHold	Panggilan API
AWS Backup	ExportBackupPlanTemplate	Panggilan API
AWS Backup gateway	TestHypervisorConfiguration	Panggilan API
AWS Billing and Cost Management	AWSPaymentInstrumentGateway.Dapatkan	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.DescribeMakePaymentPage	Aksi konsol

Layanan	Nama peristiwa	Jenis peristiwa
AWS Billing and Cost Management	AWSPaymentPortalService.DescribePaymentsDashboard	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetAccountPreferences	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetAdvancePaySummary	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetAsoBulkDownload	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetBillingContactAddress	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetDocuments	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetEligiblePaymentInstruments	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetEntitiesByIds	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetFundingDocuments	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetKybcValidationStatus	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetOneTimePasswordStatus	Aksi konsol

Layanan	Nama peristiwa	Jenis peristiwa
AWS Billing and Cost Management	AWSPaymentPortalService.GetPaymentHistory	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetPaymentProfileByArn	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetPaymentProfileCurrencies	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetPaymentProfiles	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetPaymentProfileServiceProviders	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetPaymentsDue	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetRemittanceInformation	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetTaxInvoiceMetadata	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetTermsAndConditionsForProgramGroup	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetTransactionsHistory	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPortalService.GetUnappliedFunds	Aksi konsol

Layanan	Nama peristiwa	Jenis peristiwa
AWS Billing and Cost Management	AWSPaymentPortalService.GetUnpaidInvoices	Aksi konsol
AWS Billing and Cost Management	AWSPaymentPreferenceGateway.Dapatkan	Aksi konsol
AWS Billing and Cost Management	CancelBulkDownload	Aksi konsol
AWS Billing and Cost Management	DownloadCommercialInvoice	Aksi konsol
AWS Billing and Cost Management	DownloadCsv	Aksi konsol
AWS Billing and Cost Management	DownloadDoc	Aksi konsol
AWS Billing and Cost Management	UnduhECSV ForBillingPeriod	Aksi konsol
AWS Billing and Cost Management	DownloadPaymentHistory	Aksi konsol
AWS Billing and Cost Management	DownloadRegistrationDocument	Aksi konsol
AWS Billing and Cost Management	DownloadTaxInvoice	Aksi konsol
AWS Billing and Cost Management	FindBankRedirectPaymentInstruments	Aksi konsol
AWS Billing and Cost Management	FindCSV ForBillingPeriod	Aksi konsol
AWS Billing and Cost Management	ValidateReportDestination	Aksi konsol

Layanan	Nama peristiwa	Jenis peristiwa
AWS Billing and Cost Management	VerifyChinaPaymentEligibility	Aksi konsol
Amazon Braket	SearchCompilations	Panggilan API
Amazon Braket	SearchDevices	Panggilan API
Amazon Braket	SearchQuantumTasks	Panggilan API
Kasus Amazon Connect	BatchGetField	Panggilan API
Kasus Amazon Connect	SearchCases	Panggilan API
Kasus Amazon Connect	SearchRelatedItems	Panggilan API
Amazon Chime	RetrieveDataExports	Panggilan API
Amazon Chime	SearchChannels	Panggilan API
Identitas SDK Amazon Chime	DeleteProfile	Acara layanan
Identitas SDK Amazon Chime	DeleteWorkTalkAccount	Acara layanan
AWS Kamar Bersih	BatchGetSchema	Panggilan API
Direktori Cloud Amazon	BatchRead	Panggilan API
Direktori Cloud Amazon	LookupPolicy	Panggilan API
AWS CloudFormation	DetectStackDrift	Panggilan API
AWS CloudFormation	DetectStackResourceDrift	Panggilan API
AWS CloudFormation	DetectStackSetDrift	Panggilan API
AWS CloudFormation	EstimateTemplateCost	Panggilan API
AWS CloudFormation	ValidateTemplate	Panggilan API
AWS CloudShell	RedeemCode	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
AWS CloudTrail	LookupEvents	Panggilan API
AWS CodeArtifact	ReadFromRepository	Panggilan API
AWS CodeArtifact	SearchPackages	Panggilan API
AWS CodeArtifact	VerifyResourcesExistForTagris	Panggilan API
AWS CodeBuild	BatchGetBuildBatches	Panggilan API
AWS CodeBuild	BatchGetBuilds	Panggilan API
AWS CodeBuild	BatchGetProjects	Panggilan API
AWS CodeBuild	BatchGetReportGroups	Panggilan API
AWS CodeBuild	BatchGetReports	Panggilan API
AWS CodeBuild	BatchPutCodeCoverages	Panggilan API
AWS CodeBuild	BatchPutTestCases	Panggilan API
AWS CodeBuild	RequestBadge	Acara layanan
AWS CodeCommit	BatchDescribeMergeConflicts	Panggilan API
AWS CodeCommit	BatchGetCommits	Panggilan API
AWS CodeCommit	BatchGetPullRequests	Panggilan API
AWS CodeCommit	BatchGetRepositories	Panggilan API
AWS CodeCommit	EvaluatePullRequestApprovalRules	Panggilan API
AWS CodeCommit	GitPull	Panggilan API
AWS CodeDeploy	BatchGetApplicationRevisions	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
AWS CodeDeploy	BatchGetApplications	Panggilan API
AWS CodeDeploy	BatchGetDeploymentGroups	Panggilan API
AWS CodeDeploy	BatchGetDeployment Instances	Panggilan API
AWS CodeDeploy	BatchGetDeployments	Panggilan API
AWS CodeDeploy	BatchGetDeploymentTargets	Panggilan API
AWS CodeDeploy	BatchGetOnPremises Instances	Panggilan API
Amazon CodeGuru Profiler	BatchGetFrameMetricData	Panggilan API
Amazon CodeGuru Profiler	SubmitFeedback	Panggilan API
AWS CodePipeline	PollForJobs	Panggilan API
AWS CodePipeline	PollForThirdPartyJobs	Panggilan API
CodeConnections	StartAppRegistrationHandshake	Panggilan API
CodeConnections	StarTo AuthHandshake	Panggilan API
CodeConnections	ValidateHostWebhook	Panggilan API
Amazon CodeWhisperer	CreateCodeScan	Panggilan API
Amazon CodeWhisperer	CreateProfile	Panggilan API
Amazon CodeWhisperer	CreateUploadUrl	Panggilan API
Amazon CodeWhisperer	GenerateRecommendations	Panggilan API
Amazon CodeWhisperer	UpdateProfile	Panggilan API
Identitas Amazon Cognito	LookupDeveloperIdentity	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
Kolam pengguna Amazon Cognito	AdminGetDevice	Panggilan API
Kolam pengguna Amazon Cognito	AdminGetUser	Panggilan API
Kolam pengguna Amazon Cognito	AdminListDevices	Panggilan API
Kolam pengguna Amazon Cognito	AdminListGroupsForUser	Panggilan API
Kolam pengguna Amazon Cognito	AdminListUserAuthEvents	Panggilan API
Kolam pengguna Amazon Cognito	Beta_Authorize_Dapatkan	Acara layanan
Kolam pengguna Amazon Cognito	Konfirmasi_dapatkan	Acara layanan
Kolam pengguna Amazon Cognito	ConfirmForgotPassword_DAPATKAN	Acara layanan
Kolam pengguna Amazon Cognito	Error_get	Acara layanan
Kolam pengguna Amazon Cognito	ForgotPassword_DAPATKAN	Acara layanan
Kolam pengguna Amazon Cognito	IntrospectToken	Panggilan API
Kolam pengguna Amazon Cognito	Login_error_post	Acara layanan
Kolam pengguna Amazon Cognito	Login_get	Acara layanan

Layanan	Nama peristiwa	Jenis peristiwa
Kolam pengguna Amazon Cognito	MFA_Dapatkan	Acara layanan
Kolam pengguna Amazon Cognito	MfaOption_DAPATKAN	Acara layanan
Kolam pengguna Amazon Cognito	ResetPassword_DAPATKAN	Acara layanan
Kolam pengguna Amazon Cognito	Mendaftar_Dapatkan	Acara layanan
Kolam pengguna Amazon Cognito	UserInfo_DAPATKAN	Acara layanan
Kolam pengguna Amazon Cognito	UserInfo_POSTING	Acara layanan
Amazon Cognito Sync	BulkPublish	Panggilan API
Amazon Comprehend	BatchContainsPiiEntities	Panggilan API
Amazon Comprehend	BatchDetectDominantLanguage	Panggilan API
Amazon Comprehend	BatchDetectEntities	Panggilan API
Amazon Comprehend	BatchDetectKeyPhrases	Panggilan API
Amazon Comprehend	BatchDetectPiiEntities	Panggilan API
Amazon Comprehend	BatchDetectSentiment	Panggilan API
Amazon Comprehend	BatchDetectSyntax	Panggilan API
Amazon Comprehend	BatchDetectTargetedSentiment	Panggilan API
Amazon Comprehend	ClassifyDocument	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
Amazon Comprehend	ContainsPiiEntities	Panggilan API
Amazon Comprehend	DetectDominantLanguage	Panggilan API
Amazon Comprehend	DetectEntities	Panggilan API
Amazon Comprehend	DetectKeyPhrases	Panggilan API
Amazon Comprehend	DetectPiiEntities	Panggilan API
Amazon Comprehend	DetectSentiment	Panggilan API
Amazon Comprehend	DetectSyntax	Panggilan API
Amazon Comprehend	DetectTargetedSentiment	Panggilan API
Amazon Comprehend	DetectToxicContent	Panggilan API
AWS Compute Optimizer	ExportAutoScalingGroupRecommendations	Panggilan API
AWS Compute Optimizer	EksporteB VolumeRecommendations	Panggilan API
AWS Compute Optimizer	ExporTec InstanceRecommendations	Panggilan API
AWS Compute Optimizer	EksporECS ServiceRecommendations	Panggilan API
AWS Compute Optimizer	ExportLambdaFunctionRecommendations	Panggilan API
AWS Compute Optimizer	Exportrds InstanceRecommendations	Panggilan API
AWS Config	BatchGetAggregateResourceConfig	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
AWS Config	BatchGetResourceConfig	Panggilan API
AWS Config	SelectAggregateResourceConfig	Panggilan API
AWS Config	SelectResourceConfig	Panggilan API
Amazon Connect	AdminGetEmergencyAccessToken	Panggilan API
Amazon Connect	SearchQueues	Panggilan API
Amazon Connect	SearchRoutingProfiles	Panggilan API
Amazon Connect	SearchSecurityProfiles	Panggilan API
Amazon Connect	SearchUsers	Panggilan API
AWS Glue DataBrew	SendProjectSessionAction	Panggilan API
AWS Data Pipeline	EvaluateExpression	Panggilan API
AWS Data Pipeline	QueryObjects	Panggilan API
AWS Data Pipeline	ValidatePipelineDefinition	Panggilan API
AWS DataSync	VerifyResourcesExistForTagris	Panggilan API
AWS DeepLens	BatchGetDevice	Panggilan API
AWS DeepLens	BatchGetModel	Panggilan API
AWS DeepLens	BatchGetProject	Panggilan API
AWS DeepLens	CreateDeviceCertificates	Panggilan API
AWS DeepRacer	AdminGetAccountConfig	Panggilan API
AWS DeepRacer	AdminListAssociatedUsers	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
AWS DeepRacer	TestRewardFunction	Panggilan API
AWS DeepRacer	VerifyResourcesExistForTagris	Panggilan API
Amazon Detective	BatchGetGraphMemberDatasources	Panggilan API
Amazon Detective	BatchGetMembershipDatasources	Panggilan API
Amazon Detective	SearchGraph	Panggilan API
DevOpsGuru Amazon	SearchInsights	Panggilan API
DevOpsGuru Amazon	SearchOrganizationInsights	Panggilan API
AWS Database Migration Service	BatchStartRecommendations	Panggilan API
AWS Database Migration Service	ModifyRecommendation	Panggilan API
AWS Database Migration Service	StartRecommendations	Panggilan API
AWS Database Migration Service	VerifyResourcesExistForTagris	Panggilan API
AWS Directory Service	VerifyTrust	Panggilan API
Amazon Elastic Compute Cloud	ConfirmProductInstance	Panggilan API
Amazon Elastic Compute Cloud	ReportInstanceStatus	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
Amazon Elastic Container Registry	BatchCheckLayerAvailability	Panggilan API
Amazon Elastic Container Registry	BatchGetImage	Panggilan API
Amazon Elastic Container Registry	BatchGetImageReferrer	Panggilan API
Amazon Elastic Container Registry	BatchGetRepository ScanningConfiguration	Panggilan API
Amazon Elastic Container Registry	DryRunEvent	Acara layanan
Amazon Elastic Container Registry	PolicyExecutionEvent	Acara layanan
Amazon Elastic Container Registry Publik	BatchCheckLayerAvailability	Panggilan API
Amazon Elastic Container Service	DiscoverPollEndpoint	Panggilan API
Amazon Elastic Container Service	FindSubfleetRoute	Panggilan API
Amazon Elastic Container Service	ValidateResources	Panggilan API
Amazon Elastic Container Service	VerifyTaskSetsExist	Panggilan API
Amazon Elastic Kubernetes Service	AccessKubernetesApi	Panggilan API
AWS Elastic Beanstalk	CheckDNSKetersediaan	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
AWS Elastic Beanstalk	RequestEnvironmentInfo	Panggilan API
AWS Elastic Beanstalk	RetrieveEnvironmentInfo	Panggilan API
AWS Elastic Beanstalk	ValidateConfigurationSettings	Panggilan API
Amazon Elastic File System	NewClientConnection	Acara layanan
Amazon Elastic File System	UpdateClientConnection	Acara layanan
Amazon Elastic Transcoder	ReadJob	Panggilan API
Amazon Elastic Transcoder	ReadPipeline	Panggilan API
Amazon Elastic Transcoder	ReadPreset	Panggilan API
Amazon EventBridge	TestEventPattern	Panggilan API
Amazon EventBridge	TestScheduleExpression	Panggilan API
Amazon FinSpace API	BatchListCatalogNodesByDataset	Panggilan API
Amazon FinSpace API	BatchListNodesByDataset	Panggilan API
Amazon FinSpace API	BatchValidateAccess	Panggilan API
Amazon FinSpace API	CreateAuditRecordsQuery	Panggilan API
Amazon FinSpace API	SearchDatasets	Panggilan API
Amazon FinSpace API	SearchDatasetsV	Panggilan API
Amazon FinSpace API	ValidateIdToken	Panggilan API
AWS Firewall Manager	DisassociateAdminAccount	Panggilan API
Amazon Forecast	InvokeForecastEndpoint	Panggilan API
Amazon Forecast	QueryFeature	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
Amazon Forecast	QueryForecast	Panggilan API
Amazon Forecast	QueryWhatIfForecast	Panggilan API
Amazon Forecast	VerifyResourcesExistForTagris	Panggilan API
Amazon Fraud Detector	BatchGetVariable	Panggilan API
Amazon Fraud Detector	VerifyResourcesExistForTagris	Panggilan API
FreeRTOS	VerifyEmailAddress	Panggilan API
Amazon GameLift	RequestUploadCredentials	Panggilan API
Amazon GameLift	ResolveAlias	Panggilan API
Amazon GameLift	SearchGameSessions	Panggilan API
Amazon GameLift	ValidateMatchmakingRuleSet	Panggilan API
Amazon GameSparks	ExportSnapshot	Panggilan API
Amazon Location Service	BatchGetDevicePosition	Panggilan API
Amazon Location Service	CalculateRoute	Panggilan API
Amazon Location Service	CalculateRouteMatrix	Panggilan API
Amazon Location Service	SearchPlaceIndexForPosition	Panggilan API
Amazon Location Service	SearchPlaceIndexForSuggestions	Panggilan API
Amazon Location Service	SearchPlaceIndexForText	Panggilan API
Amazon S3 Glacier	InitiateJob	Panggilan API
AWS Glue	BatchGetBlueprints	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
AWS Glue	BatchGetColumnStatisticsForTable	Panggilan API
AWS Glue	BatchGetCrawlers	Panggilan API
AWS Glue	BatchGetCustomEntityTypes	Panggilan API
AWS Glue	BatchGetDataQualityResult	Panggilan API
AWS Glue	BatchGetDevEndpoints	Panggilan API
AWS Glue	BatchGetJobs	Panggilan API
AWS Glue	BatchGetMLTransform	Panggilan API
AWS Glue	BatchGetPartition	Panggilan API
AWS Glue	BatchGetTriggers	Panggilan API
AWS Glue	BatchGetWorkflows	Panggilan API
AWS Glue	QueryJobRuns	Panggilan API
AWS Glue	QueryJobRunsAggregated	Panggilan API
AWS Glue	QueryJobs	Panggilan API
AWS Glue	QuerySchemaVersionMetadata	Panggilan API
AWS Glue	SearchTables	Panggilan API
AWS HealthLake	ReadResource	Panggilan API
AWS HealthLake	SearchWithGet	Panggilan API
AWS HealthLake	SearchWithPost	Panggilan API
AWS Identity and Access Management	GenerateCredentialReport	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
AWS Identity and Access Management	GenerateOrganizationsAccessReport	Panggilan API
AWS Identity and Access Management	GenerateServiceLastAccessedDetails	Panggilan API
AWS Identity and Access Management	SimulateCustomPolicy	Panggilan API
AWS Identity and Access Management	SimulatePrincipalPolicy	Panggilan API
AWS Toko Identitas	IsMemberInGroups	Panggilan API
AWS Auth Toko Identitas	BatchGetSession	Panggilan API
Amazon Inspector Klasik	PreviewAgents	Panggilan API
Amazon Inspector Klasik	BatchGetAccountStatus	Panggilan API
Amazon Inspector Klasik	BatchGetFreeTrialInfo	Panggilan API
Amazon Inspector Klasik	BatchGetMember	Panggilan API
AWS Invoicing	ValidateDocumentDeliveryS3LocationInfo	Panggilan API
AWS IoT	SearchIndex	Panggilan API
AWS IoT	TestAuthorization	Panggilan API
AWS IoT	TestInvokeAuthorizer	Panggilan API
AWS IoT	ValidateSecurityProfileBehaviors	Panggilan API
AWS IoT Analytics	SampleChannelData	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
AWS IoT SiteWise	GatewaysVerifyResourcesExistForTagrisInternal	Panggilan API
AWS IoT Things Graph	SearchEntities	Panggilan API
AWS IoT Things Graph	SearchFlowExecutions	Panggilan API
AWS IoT Things Graph	SearchFlowTemplates	Panggilan API
AWS IoT Things Graph	SearchSystemInstances	Panggilan API
AWS IoT Things Graph	SearchSystemTemplates	Panggilan API
AWS IoT Things Graph	SearchThings	Panggilan API
AWS IoT TwinMaker	ExecuteQuery	Panggilan API
AWS IoT Wireless	CreateNetworkAnalyzerConfiguration	Panggilan API
AWS IoT Wireless	DeleteNetworkAnalyzerConfiguration	Panggilan API
AWS IoT Wireless	DeregisterWirelessDevice	Panggilan API
Amazon Interactive Video Service	BatchGetChannel	Panggilan API
Amazon Interactive Video Service	BatchGetStreamKey	Panggilan API
Amazon Kendra	BatchGetDocumentStatus	Panggilan API
Amazon Kendra	Query	Panggilan API
Layanan Terkelola Amazon untuk Apache Flink	DiscoverInputSchema	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
AWS Key Management Service	Dekripsi	Panggilan API
AWS Key Management Service	Enkripsi	Panggilan API
AWS Key Management Service	GenerateDataKey	Panggilan API
AWS Key Management Service	GenerateDataKeyPair	Panggilan API
AWS Key Management Service	GenerateDataKeyPairWithoutPlaintext	Panggilan API
AWS Key Management Service	GenerateDataKeyWithoutPlaintext	Panggilan API
AWS Key Management Service	GenerateMac	Panggilan API
AWS Key Management Service	GenerateRandom	Panggilan API
AWS Key Management Service	ReEncrypt	Panggilan API
AWS Key Management Service	Sign	Panggilan API
AWS Key Management Service	Verifikasi	Panggilan API
AWS Key Management Service	VerifyMac	Panggilan API
AWS Lake Formation	SearchDatabasesByLFTags	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
AWS Lake Formation	SearchTablesByLFTags	Panggilan API
AWS Lake Formation	StartQueryPlanning	Panggilan API
Amazon Lex	BatchCreateCustomVocabularyItem	Panggilan API
Amazon Lex	BatchDeleteCustomVocabularyItem	Panggilan API
Amazon Lex	BatchUpdateCustomVocabularyItem	Panggilan API
Amazon Lex	DeleteCustomVocabulary	Panggilan API
Amazon Lex	SearchAssociatedTranscripts	Panggilan API
Amazon Lightsail	CreateGUI SessionAccessDetails	Panggilan API
Amazon Lightsail	DownloadDefaultKeyPair	Panggilan API
Amazon Lightsail	IsVpcPeered	Panggilan API
CloudWatch Log Amazon	FilterLogEvents	Panggilan API
Amazon Macie	BatchGetCustomDataIdentifiers	Panggilan API
Amazon Macie	UpdateFindingsFilter	Panggilan API
AWS Elemental MediaConnect	ManagedDescribeFlow	Panggilan API
AWS Elemental MediaConnect	PrivateDescribeFlowMeta	Panggilan API
AWS Application Migration Service	OperationalDescribeJobLogItems	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
AWS Application Migration Service	OperationalDescribeJobs	Panggilan API
AWS Application Migration Service	OperationalDescribeReplicationConfigurationTemplates	Panggilan API
AWS Application Migration Service	OperationalDescribeSourceServer	Panggilan API
AWS Application Migration Service	OperationalGetLaunchConfiguration	Panggilan API
AWS Application Migration Service	OperationalListSourceServers	Panggilan API
AWS Application Migration Service	VerifyClientRoleForMgn	Panggilan API
AWS HealthOmics	VerifyResourceExists	Panggilan API
AWS HealthOmics	VerifyResourcesExistForTagris	Panggilan API
Amazon Polly	SynthesizeLongSpeech	Panggilan API
Amazon Polly	SynthesizeSpeech	Panggilan API
Amazon Polly	SynthesizeSpeechGet	Panggilan API
AWS layanan yang menyediakan jaringan pribadi terkelola	Ping	Panggilan API
AWS Proton	DeleteEnvironmentTemplateVersion	Panggilan API
AWS Proton	DeleteServiceTemplateVersion	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
Amazon QLDB	ShowCatalog	Panggilan API
Amazon QuickSight	GenerateEmbedUrlFo rAnonymousUser	Panggilan API
Amazon QuickSight	GenerateEmbedUrlFo rRegisteredUser	Panggilan API
Amazon QuickSight	QueryDatabase	Acara layanan
Amazon QuickSight	SearchAnalyses	Panggilan API
Amazon QuickSight	SearchDashboards	Panggilan API
Amazon QuickSight	SearchDataSets	Panggilan API
Amazon QuickSight	SearchDataSources	Panggilan API
Amazon QuickSight	SearchFolders	Panggilan API
Amazon QuickSight	SearchGroups	Panggilan API
Amazon QuickSight	SearchUsers	Panggilan API
Amazon Relational Database Service	DownloadCompleteDB LogFile	Panggilan API
Amazon Relational Database Service	DownloAdDB LogFilePortion	Panggilan API
Amazon Rekognition	CompareFaces	Panggilan API
Amazon Rekognition	DetectCustomLabels	Panggilan API
Amazon Rekognition	DetectFaces	Panggilan API
Amazon Rekognition	DetectLabels	Panggilan API
Amazon Rekognition	DetectModerationLabels	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
Amazon Rekognition	DetectProtectiveEquipment	Panggilan API
Amazon Rekognition	DetectText	Panggilan API
Amazon Rekognition	RecognizeCelebrities	Panggilan API
Amazon Rekognition	SearchFaces	Panggilan API
Amazon Rekognition	SearchFacesByImage	Panggilan API
Amazon Rekognition	SearchUsers	Panggilan API
Amazon Rekognition	SearchUsersByImage	Panggilan API
Penjelajah Sumber Daya AWS	BatchGetView	Panggilan API
Penjelajah Sumber Daya AWS	Pencarian	Panggilan API
AWS Resource Groups	SearchResources	Panggilan API
AWS Resource Groups	ValidateResourceSharing	Panggilan API
AWS RoboMaker	BatchDescribeSimulationJob	Panggilan API
Amazon Route 53	TestDNSsAnswer	Panggilan API
Domain Amazon Route 53	CheckAvailability	Panggilan API
Domain Amazon Route 53	CheckDomainAvailability	Panggilan API
Domain Amazon Route 53	checkDomainTransferability	Panggilan API
Domain Amazon Route 53	CheckDomainTransferability	Panggilan API
Domain Amazon Route 53	isEmailReachable	Panggilan API
Domain Amazon Route 53	SearchDomains	Panggilan API
Domain Amazon Route 53	sendVerificationMessage	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
Domain Amazon Route 53	ViewBilling	Panggilan API
Domain Amazon Route 53	ViewBilling	Panggilan API
Amazon CloudWatch RUM	BatchGetRumMetricDefinitions	Panggilan API
Amazon Simple Storage Service	gema	Panggilan API
Amazon Simple Storage Service	GenerateInventory	Acara layanan
Amazon SageMaker	BatchDescribeModelPackage	Panggilan API
Amazon SageMaker	DeleteModelCard	Panggilan API
Amazon SageMaker	QueryLineage	Panggilan API
Amazon SageMaker	RenderUiTemplate	Panggilan API
Amazon SageMaker	Pencarian	Panggilan API
EventBridge Skema Amazon	ExportSchema	Panggilan API
EventBridge Skema Amazon	SearchSchemas	Panggilan API
Amazon SimpleDB	DomainMetadata	Panggilan API
AWS Secrets Manager	ValidateResourcePolicy	Panggilan API
AWS Service Catalog	ScanProvisionedProducts	Panggilan API
AWS Service Catalog	SearchProducts	Panggilan API
AWS Service Catalog	SearchProductsAsAdmin	Panggilan API
AWS Service Catalog	SearchProvisionedProducts	Panggilan API
Amazon SES	BatchGetMetricData	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
Amazon SES	TestRenderEmailTemplate	Panggilan API
Amazon SES	TestRenderTemplate	Panggilan API
Amazon Simple Notification Service	CheckIfPhoneNumber IsOptedOut	Panggilan API
AWS SQL Workbench	BatchGetNotebookCell	Panggilan API
AWS SQL Workbench	ExportNotebook	Panggilan API
Amazon EC2 Systems Manager	ExecuteApi	Panggilan API
AWS Systems Manager Incident Manager	DeleteContactChannel	Panggilan API
AWS IAM Identity Center	IsMemberInGroup	Panggilan API
AWS IAM Identity Center	SearchGroups	Panggilan API
AWS IAM Identity Center	SearchUsers	Panggilan API
AWS STS	AssumeRole	Panggilan API
AWS STS	AssumeRoleWithSAM	Panggilan API
AWS STS	AssumeRoleWithWebIdentity	Panggilan API
AWS STS	DecodeAuthorizationMessage	Panggilan API
AWS Pengaturan Pajak	BatchGetTaxExemptions	Panggilan API
AWS WAFV2	CheckCapacity	Panggilan API
AWS WAFV2	GenerateMobileSdkR eleaseUrl	Panggilan API
AWS Well-Architected Tool	ExportLens	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
AWS Well-Architected Tool	TagResource	Panggilan API
AWS Well-Architected Tool	UntagResource	Panggilan API
AWS Well-Architected Tool	UpdateGlobalSettings	Panggilan API
Kebijaksanaan Amazon Connect	QueryAssistant	Panggilan API
Kebijaksanaan Amazon Connect	SearchContent	Panggilan API
Kebijaksanaan Amazon Connect	SearchSessions	Panggilan API
Amazon WorkDocs	AbortDocumentVersionUpload	Panggilan API
Amazon WorkDocs	AddUsersToGroup	Panggilan API
Amazon WorkDocs	BatchGetUsers	Panggilan API
Amazon WorkDocs	CheckAlias	Panggilan API
Amazon WorkDocs	CompleteDocumentVersionUpload	Panggilan API
Amazon WorkDocs	CreateAnnotation	Panggilan API
Amazon WorkDocs	CreateComment	Panggilan API
Amazon WorkDocs	CreateFeedbackRequest	Panggilan API
Amazon WorkDocs	CreateFolder	Panggilan API
Amazon WorkDocs	CreateGroup	Panggilan API
Amazon WorkDocs	CreateShare	Panggilan API
Amazon WorkDocs	CreateUser	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
Amazon WorkDocs	DeleteAnnotation	Panggilan API
Amazon WorkDocs	DeleteComment	Panggilan API
Amazon WorkDocs	DeleteDocument	Panggilan API
Amazon WorkDocs	DeleteFeedbackRequest	Panggilan API
Amazon WorkDocs	DeleteFolder	Panggilan API
Amazon WorkDocs	DeleteFolderContents	Panggilan API
Amazon WorkDocs	DeleteGroup	Panggilan API
Amazon WorkDocs	DeleteOrganizationShare	Panggilan API
Amazon WorkDocs	DeleteUser	Panggilan API
Amazon WorkDocs	DownloadDocumentVersion	Panggilan API
Amazon WorkDocs	DownloadDocumentVersionUnderlays	Panggilan API
Amazon WorkDocs	InitiateDocumentVersionUpload	Panggilan API
Amazon WorkDocs	LogoutUser	Panggilan API
Amazon WorkDocs	PaginatedOrganizationActivity	Panggilan API
Amazon WorkDocs	PublishAnnotations	Panggilan API
Amazon WorkDocs	PublishComments	Panggilan API
Amazon WorkDocs	RestoreDocument	Panggilan API
Amazon WorkDocs	RestoreFolder	Panggilan API
Amazon WorkDocs	SearchGroups	Panggilan API

Layanan	Nama peristiwa	Jenis peristiwa
Amazon WorkDocs	SearchOrganizationUsers	Panggilan API
Amazon WorkDocs	TransferUserResources	Panggilan API
Amazon WorkDocs	UpdateAnnotation	Panggilan API
Amazon WorkDocs	UpdateComment	Panggilan API
Amazon WorkDocs	UpdateDocument	Panggilan API
Amazon WorkDocs	UpdateDocumentVersion	Panggilan API
Amazon WorkDocs	UpdateFolder	Panggilan API
Amazon WorkDocs	UpdateGroup	Panggilan API
Amazon WorkDocs	UpdateOrganization	Panggilan API
Amazon WorkDocs	UpdateUser	Panggilan API
Amazon WorkMail	AssumeImpersonationRole	Panggilan API
Amazon WorkMail	QueryDnsRecords	Panggilan API
Amazon WorkMail	SearchMembers	Panggilan API
Amazon WorkMail	TestAvailabilityConfiguration	Panggilan API
Amazon WorkMail	TestInboundMailFlowRules	Panggilan API
Amazon WorkMail	TestOutboundMailFlowRules	Panggilan API

EventBridge referensi detail acara

EventBridge sendiri memancarkan peristiwa-peristiwa berikut. Peristiwa ini secara otomatis dikirim ke bus acara default seperti halnya AWS layanan lainnya.

Untuk definisi bidang metadata yang disertakan dalam semua peristiwa, lihat. the section called “Referensi struktur acara”

Topik

- Acara Terjadwal
- Skema Dibuat
- Versi Skema Dibuat

Acara Terjadwal

Di bawah ini adalah bidang detail untuk `Scheduled Event` acara tersebut.

`detail-type`Bidang `source` dan disertakan karena mengandung nilai khusus untuk EventBridge acara. Untuk definisi bidang metadata lain yang disertakan dalam semua peristiwa, lihat. the section called “Referensi struktur acara”

```
{
  . . .,
  "detail-type": "Scheduled Event",
  "source": "aws.events",
  . . .,
  "detail": {}
}
```

`detail-type`

Mengidentifikasi jenis acara.

Untuk acara ini, nilai ini adalah`Scheduled Event`.

Diperlukan: Ya

`source`

Mengidentifikasi layanan yang menghasilkan peristiwa. Untuk EventBridge acara, nilai ini adalah`aws.events`.

Diperlukan: Ya

`detail`

Objek JSON yang berisi informasi tentang peristiwa. Layanan yang menghasilkan acara menentukan konten bidang ini.

Diperlukan: Ya

Tidak ada bidang wajib dalam objek ini untuk `Scheduled Event` acara.

Example Contoh acara Acara Terjadwal

```
{
  "version": "0",
  "id": "89d1a02d-5ec7-412e-82f5-13505f849b41",
  "detail-type": "Scheduled Event",
  "source": "aws.events",
  "account": "123456789012",
  "time": "2016-12-30T18:44:49Z",
  "region": "us-east-1",
  "resources": ["arn:aws:events:us-east-1:123456789012:rule/SampleRule"],
  "detail": {}
}
```

Skema Dibuat

Di bawah ini adalah bidang detail untuk `Schema Created` acara tersebut.

Saat skema dibuat, EventBridge kirimkan a `Schema Created` dan `Schema Version Created` acara.

`detail-type`Bidang `source` dan disertakan karena mengandung nilai khusus untuk EventBridge acara. Untuk definisi bidang metadata lain yang disertakan dalam semua peristiwa, lihat. the section called “Referensi struktur acara”

```
{
  . . .,
  "detail-type": "Schema Created",
  "source": "aws.schemas",
  . . .,
  "detail": {
    "SchemaName" : "String",
    "SchemaType" : "String",
    "RegistryName" : "String",
    "CreationDate" : "DateTime",
    "Version" : "Number"
  }
```

```
}
```

`detail-type`

Mengidentifikasi jenis acara.

Untuk acara ini, nilai ini adalah`Schema Created`.

Diperlukan: Ya

`source`

Mengidentifikasi layanan yang menghasilkan peristiwa. Untuk EventBridge acara, nilai ini adalah`aws.schemas`.

Diperlukan: Ya

`detail`

Objek JSON yang berisi informasi tentang peristiwa. Layanan yang menghasilkan acara menentukan konten bidang ini.

Diperlukan: Ya

Untuk acara ini, data ini meliputi:

`SchemaName`

Nama skema.

Diperlukan: Ya

`SchemaType`

Tipe skema.

Nilai yang valid: `OpenApi3` | `JSONSchemaDraft4`

Diperlukan: Ya

`RegistryName`

Nama registri yang berisi skema.

Diperlukan: Ya

`CreationDate`

Tanggal skema dibuat.

Diperlukan: Ya

`Version`

Versi skema.

Untuk `Schema Created` acara, nilai ini akan selalu`1`.

Diperlukan: Ya

Example Contoh Skema acara Dibuat

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "Schema Created",
  "source": "aws.schemas",
  "account": "123456789012",
  "time": "2019-05-31T21:49:54Z",
  "region": "us-east-1",
  "resources": ["arn:aws:schemas:us-east-1::schema/myRegistry/mySchema"],
  "detail": {
    "SchemaName": "mySchema",
    "SchemaType": "OpenApi3",
    "RegistryName": "myRegistry",
    "CreationDate": "2019-11-29T20:08:55Z",
    "Version": "1"
  }
}
```

Versi Skema Dibuat

Di bawah ini adalah bidang detail untuk `Schema Version Created` acara tersebut.

Saat skema dibuat, EventBridge kirimkan a `Schema Created` dan `Schema Version Created` acara.

`detail-type`Bidang `source` dan disertakan karena mengandung nilai khusus untuk EventBridge acara. Untuk definisi bidang metadata lain yang disertakan dalam semua peristiwa, lihat. the section called “Referensi struktur acara”

```
{
```

```
  . . .,
  "detail-type": "Schema Version Created",
  "source": "aws.schemas",
  . . .,
  "detail": {
    "SchemaName" : "String",
    "SchemaType" : "String",
    "RegistryName" : "String",
    "CreationDate" : "DateTime",
    "Version" : "Number"
  }
}
```

`detail-type`

Mengidentifikasi jenis acara.

Untuk acara ini, nilai ini adalah`Schema Version Created`.

Diperlukan: Ya

`source`

Mengidentifikasi layanan yang menghasilkan peristiwa. Untuk EventBridge acara, nilai ini adalah`aws.schemas`.

Diperlukan: Ya

`detail`

Objek JSON yang berisi informasi tentang peristiwa. Layanan yang menghasilkan acara menentukan konten bidang ini.

Diperlukan: Ya

Untuk acara ini, data ini meliputi:

`SchemaName`

Nama skema.

Diperlukan: Ya

`SchemaType`

Tipe skema.

Nilai yang valid: `OpenApi3` | `JSONSchemaDraft4`

Diperlukan: Ya

`RegistryName`

Nama registri yang berisi skema.

Diperlukan: Ya

`CreationDate`

Tanggal versi skema dibuat.

Diperlukan: Ya

`Version`

Versi skema.

Diperlukan: Ya

Example Contoh Skema Versi Dibuat acara

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "Schema Version Created",
  "source": "aws.schemas",
  "account": "123456789012",
  "time": "2019-05-31T21:49:54Z",
  "region": "us-east-1",
  "resources": ["arn:aws:schemas:us-east-1::schema/myRegistry/mySchema"],
  "detail": {
    "SchemaName": "mySchema",
    "SchemaType": "OpenApi3",
    "RegistryName": "myRegistry",
    "CreationDate": "2019-11-29T20:08:55Z",
    "Version": "5"
  }
}
```

Menerima acara dari mitra SaaS dengan Amazon EventBridge

Untuk menerimaperistiwa dari aplikasi dan layanan mitra SaaS, Anda memerlukan sumber peristiwa mitra dari mitra. Kemudian Anda dapat membuat mitra bus peristiwadan mencocokkannya dengan sumber peristiwa mitra.

Video berikut mencakup integrasi SaaS dengan EventBridge: Perangkat lunak sebagai layanan (SaaS) mitra

Topik

- Integrasi mitra SaaS yang didukung
- Mengonfigurasi Amazon EventBridge untuk menerima peristiwa dari integrasi SaaS
- Membuat aturan yang cocok dengan peristiwa mitra SaaS
- Menerima acara menggunakan URL AWS Lambda fungsi
- Menerima acara dari Salesforce

Integrasi mitra SaaS yang didukung

EventBridge mendukung integrasi mitra SaaS berikut:

- Adobe
- Auth0
- Blitline
- BUIDLHub
- Buildkite
- CleverTap
- Datadog
- Epsagon
- Freshworks
- Genesys
- GS2
- Karte
- Kloudless

- Mackerel
- MongoDB
- New Relic
- OneLogin
- Opsgenie
- PagerDuty
- Payshield
- SaaSus Platform
- SailPoint
- Saviynt
- Segment
- Shopify
- SignalFx
- Site24x7
- Stax
- Stripe
- SugarCRM
- SugarCRM
- Symantec
- Thundra
- TriggerMesh
- Whispir
- Zendesk
- API Mitra Penjual Amazon

Sumber peristiwa mitra tersedia di Wilayah berikut.

Kode	Nama
`us-east-1`	US East (N. Virginia)
`us-east-2`	AS Timur (Ohio)

Kode	Nama
us-west-1	AS Barat (California Utara)
us-west-2	AS Barat (Oregon)
ca-central-1	Canada (Central)
eu-central-1	Eropa (Frankfurt)
eu-central-2	Eropa (Zürich)
eu-west-1	Eropa (Irlandia)
eu-west-2	Eropa (London)
eu-west-3	Eropa (Paris)
eu-north-1	Eropa (Stockholm)
eu-south-1	Eropa (Milan)
eu-south-2	Eropa (Spanyol)
af-south-1	Afrika (Cape Town)
ap-south-1	Asia Pasifik (Mumbai)
ap-south-2	Asia Pasifik (Hyderabad)
ap-east-1	Asia Pasifik (Hong Kong)
ap-northeast-1	Asia Pacific (Tokyo)
ap-northeast-2	Asia Pasifik (Seoul)
ap-northeast-3	Asia Pasifik (Osaka)
ap-southeast-1	Asia Pasifik (Singapura)
ap-southeast-2	Asia Pasifik (Sydney)

Kode	Nama
ap-southeast-3	Asia Pasifik (Jakarta)
ap-southeast-4	Asia Pasifik (Melbourne)
cn-north-1	China (Beijing)
cn-northwest-1	China (Ningxia)
me-central-1	Timur Tengah (UEA)
me-south-1	Timur Tengah (Bahrain)
sa-east-1	Amerika Selatan (Sao Paulo)
il-central-1	Israel (Tel Aviv)

Mengonfigurasi Amazon EventBridge untuk menerima peristiwa dari integrasi SaaS

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Sumber peristiwa mitra.
3. Temukan mitra yang Anda inginkan dan kemudian pilih Atur untuk mitra itu.
4. Untuk menyalin ID akun Anda ke clipboard, pilih Salin.
5. Di panel navigasi, pilih Sumber peristiwa mitra.
6. Kunjungi situs web mitra dan ikuti petunjuk untuk membuat sumber peristiwa mitra menggunakan ID akun Anda. Sumber peristiwa yang Anda buat hanya tersedia untuk akun Anda.
7. Kembali ke EventBridge konsol dan pilih Sumber acara Partner di panel navigasi.
8. Pilih tombol di samping sumber peristiwa mitra, lalu pilih Kaitkan dengan bus peristiwa.

   Status sumber peristiwa berubah dari `Pending` ke `Active`, dan nama bus peristiwa diperbarui agar cocok dengan nama sumber peristiwa mitra. Sekarang Anda dapat mulai membuat aturan yang cocok dengan peristiwa dari sumber peristiwa mitra. Untuk informasi selengkapnya, lihat Membuat aturan yang cocok dengan peristiwa mitra SaaS.

> **Note**
>
> Setiap acara yang diterbitkan oleh mitra ke sumber acara mitra yang belum dikaitkan dengan bus acara akan segera dijatuhkan. Peristiwa itu tidak akan bertahan saat istirahat. EventBridge

Membuat aturan yang cocok dengan peristiwa mitra SaaS

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan.

   Aturan tidak boleh memiliki nama yang sama dengan aturan lain di Wilayah yang sama dan di bus kejadian yang sama.
5. Untuk bus acara, pilih bus acara yang ingin Anda kaitkan dengan aturan ini. Jika Anda ingin aturan ini cocok dengan acara yang berasal dari akun Anda, pilih bus acara AWS default. Saat layanan AWS di akun Anda menghasilkan kejadian, layanan tersebut akan selalu masuk ke bus kejadian default akun Anda.
6. Untuk Tipe aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.
8. Untuk sumber acara, pilih Lainnya.
9. (Opsional) Untuk contoh acara, pilih jenis acara.
10. Untuk pola Peristiwa, masukkan pola acara JSON.
11. Pilih Berikutnya.
12. Untuk Jenis target, pilih Layanan AWS .
13. Untuk Pilih target, pilih AWS layanan yang ingin Anda kirimi informasinya saat EventBridge mendeteksi peristiwa yang cocok dengan pola acara.
14. Bidang yang ditampilkan bervariasi tergantung pada layanan yang Anda pilih. Masukkan informasi khusus untuk jenis target ini sesuai kebutuhan.

15. Untuk banyak jenis target, EventBridge perlu izin untuk mengirim acara ke target. Dalam kasus ini, EventBridge dapat membuat peran IAM yang diperlukan agar aturan Anda berjalan. Lakukan salah satu dari langkah berikut ini:

    - Untuk membuat IAM role secara otomatis, pilih Buat peran baru untuk sumber daya khusus ini.
    - Untuk menggunakan peran IAM yang Anda buat sebelumnya, pilih Gunakan peran yang ada dan pilih peran yang ada dari daftar drop-down.

16. (Opsional) Untuk pengaturan tambahan, lakukan hal berikut:

    a. Untuk Masa peristiwa maksimal, masukkan nilai antara satu menit (00:01) dan 24 jam (24:00).

    b. Untuk Upaya coba lagi, masukkan angka antara 0 dan 185.

    c. Untuk antrian Dead-letter, pilih apakah akan menggunakan antrean Amazon SQS standar sebagai antrian huruf mati. EventBridge mengirimkan peristiwa yang cocok dengan aturan ini ke antrian huruf mati jika tidak berhasil dikirim ke target. Lakukan salah satu hal berikut:

        - Pilih Tidak ada untuk tidak menggunakan antrean surat mati.
        - Pilih Pilihan antrean Amazon SQS di akun AWS saat ini untuk digunakan sebagai antrean surat mati kemudian pilih antrean yang akan digunakan dari daftar menurun.
        - Pilih Pilih antrean Amazon SQS di AWS akun lain sebagai antrian huruf mati dan kemudian masukkan ARN antrian yang akan digunakan. Anda harus melampirkan kebijakan berbasis sumber daya ke antrian yang memberikan EventBridge izin untuk mengirim pesan ke antrean tersebut. Untuk informasi selengkapnya, lihat Memberikan izin untuk antrean surat mati.

17. (Opsional) Pilih Tambahkan target lain untuk menambahkan target lain untuk aturan ini.
18. Pilih Berikutnya.
19. (Opsional) Masukkan satu atau lebih tanda untuk aturan. Untuk informasi selengkapnya, lihat EventBridge Tag Amazon.
20. Pilih Berikutnya.
21. Tinjau detail aturan dan pilih Buat aturan.

Menerima acara menggunakan URL AWS Lambda fungsi

> **Note**
>
> Agar Webhook Masuk dapat diakses oleh mitra kami, kami membuat Lambda Terbuka di AWS akun Anda yang diamankan di tingkat aplikasi Lambda dengan memverifikasi tanda tangan otentikasi yang dikirim oleh mitra pihak ketiga. Harap tinjau konfigurasi ini dengan tim keamanan Anda. Untuk informasi selengkapnya, lihat Model keamanan dan autentikasi untuk URL fungsi Lambda.

Bus EventBridge acara Amazon Anda dapat menggunakan URL AWS Lambda fungsi yang dibuat oleh AWS CloudFormation templat untuk menerima acara dari penyedia SaaS yang didukung. Dengan URL fungsi, data peristiwa dikirim ke fungsi Lambda. Fungsi kemudian mengubah data ini menjadi peristiwa yang dapat dicerna oleh EventBridge dan dikirim ke bus acara untuk diproses. Setelah acara berada di bus acara, Anda dapat menggunakan aturan untuk memfilter peristiwa, menerapkan transformasi input yang dikonfigurasi, dan kemudian merutekkannya ke target yang benar.

> **Note**
>
> Membuat URL fungsi Lambda akan meningkatkan biaya bulanan Anda. Untuk informasi selengkapnya, lihat harga AWS Lambda.

Untuk mengatur koneksi EventBridge, Anda terlebih dahulu memilih penyedia SaaS yang ingin Anda atur koneksi dengannya. Kemudian, Anda memberikan rahasia penandatanganan yang telah Anda buat dengan penyedia itu, dan pilih bus EventBridge acara untuk mengirim acara. Akhirnya, Anda menggunakan AWS CloudFormation template dan membuat sumber daya yang dibutuhkan untuk menyelesaikan koneksi.

Penyedia SaaS berikut saat ini tersedia untuk digunakan dengan menggunakan URL fungsi EventBridge Lambda:

- GitHub
- Stripe
- Twilio

Topik

- Siapkan koneksi ke GitHub
- Langkah 1: Buat AWS CloudFormation tumpukan
- Langkah 2: Buat GitHub webhook
- Mengatur koneksi ke Stripe
- Mengatur koneksi ke Twilio
- Perbarui rahasia webhook atau token autentikasi
- Perbarui fungsi Lambda
- Jenis acara yang tersedia
- Kuota, kode kesalahan, dan mencoba kembali pengiriman

Siapkan koneksi ke GitHub

Langkah 1: Buat AWS CloudFormation tumpukan

Pertama, gunakan EventBridge konsol Amazon untuk membuat CloudFormation tumpukan:

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Dari panel navigasi, pilih Mulai cepat.
3. Di bawah Webhook masuk menggunakan Lambda FURLs, pilih Memulai.
4. Di bawah GitHub, pilih Siapkan.
5. Di bawah Langkah 1: Pilih bus acara, pilih bus acara dari daftar dropdown. Bus acara ini menerima data dari URL fungsi Lambda yang Anda berikan. GitHub Anda juga dapat membuat bus acara dengan memilih bus acara baru.
6. Di bawah Langkah 2: Siapkan menggunakan CloudFormation, pilih GitHubWebhook baru.
7. Pilih Saya mengakui bahwa Webhook Masuk yang saya buat akan dapat diakses publik. dan pilih Konfirmasi.
8. Masukkan nama untuk tumpukan.
9. Di bawah parameter, verifikasi bahwa bus peristiwa yang benar terdaftar, lalu tentukan token aman untuk GitHubWebhookSecret. Untuk informasi selengkapnya tentang membuat token aman, lihat Menyetel token rahasia Anda di GitHub dokumentasi.
10. Di bawah Kemampuan dan transformasi, pilih masing-masing dari berikut ini:

    - Saya mengakui bahwa AWS CloudFormation mungkin menciptakan sumber daya IAM.

- Saya mengakui bahwa AWS CloudFormation mungkin membuat sumber daya IAM dengan nama khusus.
- Saya mengakui bahwa AWS CloudFormation mungkin memerlukan kemampuan berikut: **`CAPABILITY_AUTO_EXPAND`**

11. Pilih Buat tumpukan.

Langkah 2: Buat GitHub webhook

Selanjutnya, buat webhook diGitHub. Anda memerlukan token aman dan URL fungsi Lambda yang Anda buat di langkah 2 untuk menyelesaikan langkah ini. Untuk informasi selengkapnya, lihat Membuat webhook di dokumentasi. GitHub

Mengatur koneksi ke Stripe

Langkah 1: Buat titik Stripe akhir

Untuk mengatur koneksi antara EventBridge dan Stripe, pertama buat titik akhir Stripe dan catat rahasia endpoint. Anda akan menggunakan rahasia endpoint ini ketika Anda mengatur tumpukan Anda di langkah 2. Untuk informasi selengkapnya, lihat Interactive webhook endpoint builder dalam dokumentasi. Stripe

> **Note**
>
> Anda akan memerlukan URL dummy untuk mengatur titik akhir dengan. Stripe Misalnya, `www.example.com`.

Langkah 2: Buat AWS CloudFormation tumpukan

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Mulai cepat.
3. Di bawah Webhook masuk menggunakan Lambda FURLs, pilih Memulai.
4. Di bawah Stripe, pilih Siapkan.
5. Di bawah Langkah 1: Pilih dan bus acara, pilih bus acara dari daftar dropdown. Bus acara ini menerima data dari URL fungsi Lambda yang Anda berikan. Stripe Anda juga dapat membuat bus acara dengan memilih bus acara baru.
6. Di bawah Langkah 2: Siapkan menggunakan CloudFormation, pilih StripeWebhook baru.

7. Pilih Saya mengakui bahwa Webhook Masuk yang saya buat akan dapat diakses publik. dan pilih Konfirmasi.
8. Masukkan nama untuk tumpukan.
9. Di bawah parameter, verifikasi bahwa bus acara yang benar terdaftar, lalu masukkan StripeWebhookSecretyang Anda buat di Langkah 1.
10. Di bawah Kemampuan dan transformasi, pilih masing-masing dari berikut ini:
    - Saya mengakui bahwa AWS CloudFormation mungkin menciptakan sumber daya IAM.
    - Saya mengakui bahwa AWS CloudFormation mungkin membuat sumber daya IAM dengan nama khusus.
    - Saya mengakui bahwa AWS CloudFormation mungkin memerlukan kemampuan berikut: **`CAPABILITY_AUTO_EXPAND`**
11. Pilih Buat tumpukan.

Langkah 3: Perbarui titik Stripe akhir

Sekarang setelah Anda membuat URL fungsi Lambda, perbarui Stripe titik akhir untuk mengirim peristiwa ke URL fungsi Lambda.

Mengatur koneksi ke Twilio

Langkah 1: Temukan token Twilio autentikasi Anda

Untuk mengatur koneksi antara Twilio dan EventBridge, pertama-tama atur koneksi Twilio dengan token autentikasi, atau rahasia, untuk Twilio akun Anda. Untuk informasi selengkapnya, lihat Token Auth dan Cara Mengubahnya di Twilio dokumentasi.

Langkah 2: Buat AWS CloudFormation tumpukan

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Mulai cepat.
3. Di bawah Webhook masuk menggunakan Lambda FURLs, pilih Memulai.
4. Di bawah Twilio, pilih Siapkan.
5. Di bawah Langkah 1: Pilih dan bus acara, spilih bus acara dari daftar dropdown. Bus acara ini menerima data dari URL fungsi Lambda yang Anda berikan. Twilio Anda juga dapat membuat bus acara dengan memilih bus acara baru.
6. Di bawah Langkah 2: Siapkan menggunakan CloudFormation, pilih TwilioWebhook baru.

7. Pilih Saya mengakui bahwa Webhook Masuk yang saya buat akan dapat diakses publik. dan pilih Konfirmasi.
8. Masukkan nama untuk tumpukan.
9. Di bawah parameter, verifikasi bahwa bus acara yang benar terdaftar, lalu masukkan TwilioWebhookSecretyang Anda buat di Langkah 1.
10. Di bawah Kemampuan dan transformasi, pilih masing-masing dari berikut ini:
    - Saya mengakui bahwa AWS CloudFormation mungkin menciptakan sumber daya IAM.
    - Saya mengakui bahwa AWS CloudFormation mungkin membuat sumber daya IAM dengan nama khusus.
    - Saya mengakui bahwa AWS CloudFormation mungkin memerlukan kemampuan berikut: CAPABILITY_AUTO_EXPAND
11. Pilih Buat tumpukan.

Langkah 3: Buat Twilio webhook

Setelah Anda mengatur URL fungsi Lambda, Anda harus memberikannya ke Twilio sehingga data acara dapat dikirim. Untuk informasi selengkapnya, lihat Mengonfigurasi URL publik Anda dengan Twilio Twilio dokumentasi.

Perbarui rahasia webhook atau token autentikasi

Perbarui GitHub rahasia

> **Note**
>
> GitHubtidak mendukung memiliki dua rahasia pada saat yang sama. Anda mungkin mengalami downtime sumber daya sementara GitHub rahasia dan rahasia di AWS CloudFormation tumpukan tidak sinkron. GitHubpesan yang dikirim saat rahasia tidak sinkron akan gagal karena tanda tangan yang salah. Tunggu sampai CloudFormation rahasia GitHub dan sinkron, lalu coba lagi.

1. Buat GitHub rahasia baru. Untuk informasi selengkapnya, lihat Rahasia terenkripsi dalam dokumentasi. GitHub
2. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.

3. Dari panel navigasi, pilih Stacks.
4. Pilih tumpukan untuk webhook yang menyertakan rahasia yang ingin Anda perbarui.
5. Pilih Perbarui.
6. Pastikan Gunakan template saat ini dipilih dan pilih Berikutnya.
7. Di bawah GitHubWebhookSecret, hapus Gunakan nilai yang ada, masukkan GitHub rahasia baru yang Anda buat di langkah 1, dan pilih Berikutnya.
8. Pilih Berikutnya.
9. Pilih Perbarui tumpukan.

Mungkin perlu waktu hingga satu jam agar rahasia menyebar. Untuk mengurangi waktu henti ini, Anda dapat menyegarkan konteks eksekusi Lambda.

Perbarui Stripe rahasia

1. Dari Stripe dasbor, di bagian Webhooks, pilih Roll secret dan tunda kedaluwarsa setidaknya selama dua (2) jam. Untuk informasi selengkapnya, lihat Menggulung rahasia titik akhir dalam Stripe dokumentasi.
2. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
3. Dari panel navigasi, pilih Stacks.
4. Pilih tumpukan untuk webhook yang menyertakan rahasia yang ingin Anda perbarui.
5. Pilih Perbarui.
6. Pastikan Gunakan template saat ini dipilih dan pilih Berikutnya.
7. Di bawah StripeWebhookSecret, hapus Gunakan nilai yang ada, masukkan Stripe rahasia baru yang Anda buat di langkah 1, dan pilih Berikutnya.
8. Pilih Berikutnya.
9. Pilih Perbarui tumpukan.

Stripeakan mengirimkan tanda tangan lama dan tanda tangan baru selama periode rotasi.

Perbarui Twilio rahasia

> **Note**
>
> Twiliotidak mendukung memiliki dua rahasia pada saat yang sama. Anda mungkin mengalami downtime sumber daya sementara Twilio rahasia dan rahasia di AWS CloudFormation

tumpukan tidak sinkron. Twiliopesan yang dikirim saat rahasia tidak sinkron akan gagal karena tanda tangan yang salah. Tunggu sampai CloudFormation rahasia Twilio dan sinkron, lalu coba lagi.

1. Buat Twilio rahasia baru. Untuk informasi selengkapnya, lihat Token Auth dan Cara Mengubahnya di Twilio dokumentasi.
2. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
3. Dari panel navigasi, pilih Stacks.
4. Pilih tumpukan untuk webhook yang menyertakan rahasia yang ingin Anda perbarui.
5. Pilih Perbarui.
6. Pastikan Gunakan template saat ini dipilih dan pilih Berikutnya.
7. Di bawah TwilioWebhookSecret, hapus Gunakan nilai yang ada, masukkan Twilio rahasia baru yang Anda buat di langkah 1, dan pilih Berikutnya.
8. Pilih Berikutnya.
9. Pilih Perbarui tumpukan.

Mungkin perlu waktu hingga satu jam agar rahasia menyebar. Untuk mengurangi waktu henti ini, Anda dapat menyegarkan konteks eksekusi Lambda.

Perbarui fungsi Lambda

Fungsi Lambda yang dibuat oleh CloudFormation tumpukan menciptakan webhook dasar. Jika Anda ingin menyesuaikan fungsi Lambda untuk kasus penggunaan tertentu, seperti pencatatan khusus, gunakan CloudFormation konsol untuk mengakses fungsi dan kemudian gunakan konsol Lambda untuk memperbarui kode fungsi Lambda.

Akses fungsi Lambda

1. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
2. Dari panel navigasi, pilih Stacks.
3. Pilih tumpukan untuk webhook yang menyertakan fungsi Lambda yang ingin Anda perbarui.
4. Pilih tab Sumber Daya.
5. Untuk membuka fungsi Lambda di konsol Lambda, di bawah ID Fisik, pilih ID fungsi Lambda.

Sekarang setelah Anda mengakses fungsi Lambda, gunakan konsol Lambda untuk memperbarui kode fungsi.

Perbarui kode fungsi Lambda

1. Di bawah Tindakan, pilih fungsi Ekspor.
2. Pilih Unduh paket penyebaran dan simpan file ke komputer Anda.
3. Buka zip paket penyebaran file.zip, perbarui `app.py` file, dan zip paket penyebaran yang diperbarui, pastikan semua file dalam file.zip asli disertakan.
4. Di konsol Lambda, pilih tab Kode.
5. Di bagian Sumber kode, pilih Unggah dari.
6. Pilih file.zip, lalu pilih Unggah.
   - Di pemilih file, pilih file yang Anda perbarui, pilih Buka, lalu pilih Simpan.
7. Di bawah Tindakan, pilih Publikasikan versi baru.

Jenis acara yang tersedia

Jenis acara berikut saat ini didukung oleh bus CloudFormation acara:

- GitHub— Semua jenis acara didukung.
- Stripe - Semua jenis acara didukung.
- Twilio - Webhook pasca-acara didukung.

Kuota, kode kesalahan, dan mencoba kembali pengiriman

Kuota

Jumlah permintaan masuk ke webhook dibatasi oleh layanan yang mendasarinya. AWS Tabel berikut mencakup kuota yang relevan.

Layanan	Kuota
AWS Lambda	Default: 10 eksekusi bersamaan Untuk informasi selengkapnya tentang kuota, termasuk meminta kenaikan kuota, lihat Kuota Lambda.

Layanan	Kuota
AWS Secrets Manager	Default: 5.000 permintaan per detik Untuk informasi selengkapnya tentang kuota, termasuk meminta kenaikan kuota, lihat kuota.AWS Secrets Manager **Note** Jumlah permintaan per detik diminimalkan menggunakan klien caching AWS Secrets Manager Python.
Amazon EventBridge	Ukuran entri maksimum 256KB untuk PutEvents tindakan. EventBridge memberlakukan kuota tarif berbasis wilayah. Untuk informasi selengkapnya, lihat ???.

Kode eror

Setiap AWS layanan mengembalikan kode kesalahan tertentu ketika kesalahan terjadi. Tabel berikut mencakup kode kesalahan yang relevan.

Layanan	Kode kesalahan	Deskripsi
AWS Lambda	429 “” TooManyRequestsExp tion	Kuota eksekusi bersamaan terlampaui.
AWS Secrets Manager	500 “Kesalahan Server Internal”	Kuota permintaan per detik terlampaui.
Amazon EventBridge	500 “Kesalahan Server Internal”	Kuota tarif terlampaui untuk Wilayah.

Pengiriman ulang acara

Ketika kesalahan terjadi, Anda dapat mencoba kembali pengiriman peristiwa yang terpengaruh. Setiap penyedia SaaS memiliki prosedur coba ulang yang berbeda.

GitHub

Gunakan API GitHub webhooks untuk memeriksa status pengiriman panggilan webhook apa pun dan mengirimkan ulang acara, jika diperlukan. Untuk informasi selengkapnya, lihat GitHub dokumentasi berikut:

- Organisasi — Mengirimkan ulang pengiriman untuk webhook organisasi
- Repositori - Mengirimkan ulang pengiriman untuk webhook repositori
- App — Mengirimkan ulang pengiriman untuk webhook aplikasi

Stripe

Stripemencoba mengirimkan webhook Anda hingga tiga hari dengan mundur eksponensial. Untuk informasi selengkapnya, lihat Stripe dokumentasi berikut:

- Upaya pengiriman dan percobaan ulang
- Menangani kesalahan

Twilio

Twiliopengguna dapat menyesuaikan opsi coba lagi acara menggunakan penggantian koneksi. Untuk informasi selengkapnya, lihat Webhooks (callback HTTP): Penggantian Koneksi dalam dokumentasi. Twilio

Menerima acara dari Salesforce

Anda dapat menggunakan Amazon EventBridge untuk menerima acara dengan Salesforce cara berikut:

- Dengan menggunakan fitur Salesforce's Event Bus Relay untuk menerima acara langsung di bus acara EventBridge mitra.
- Dengan mengonfigurasi aliran di Amazon AppFlow yang digunakan Salesforce sebagai sumber data. Amazon AppFlow kemudian mengirimkan Salesforce acara ke EventBridge dengan menggunakan bus acara mitra.

Anda dapat mengirim informasi acara untuk Salesforce menggunakan tujuan API. Setelah acara dikirim keSalesforce, itu dapat diproses oleh pemicu Flows atau Apex. Untuk informasi selengkapnya tentang menyiapkan tujuan Salesforce API, lihat???.

Topik

- Menerima acara dari Salesforce menggunakan Event Bus Relay
- Menerima acara dari Salesforce menggunakan Amazon AppFlow

Menerima acara dari Salesforce menggunakan Event Bus Relay

Langkah 1: Siapkan Salesforce Event Bus Relay dan sumber acara EventBridge mitra

Saat Anda membuat konfigurasi relai peristiwa aktifSalesforce, Salesforce buat sumber acara mitra EventBridge dalam status tertunda.

Untuk mengkonfigurasi Salesforce Event Bus Relay

1. Siapkan Alat API REST
2. (Opsional) Tentukan Acara Platform
3. Membuat Saluran untuk Acara Platform Kustom
4. Buat Anggota Saluran untuk Mengaitkan Acara Platform Kustom
5. Buat Kredensyal Bernama
6. Buat Konfigurasi Relay Acara

Langkah 2: Aktifkan sumber acara Salesforce mitra di EventBridge konsol dan mulai relai acara

1. Buka halaman Sumber acara Partner di EventBridge konsol.
2. Pilih sumber acara Salesforce mitra yang Anda buat di Langkah 1.
3. Pilih Berasosiasi dengan bus peristiwa.
4. Memvalidasi nama bus peristiwa mitra.
5. Pilih Kaitkan.
6. Mulai Relay Acara

Setelah menyiapkan dan memulai Relay Bus Acara dan mengonfigurasi sumber acara mitra, Anda dapat membuat EventBridge aturan yang bereaksi terhadap peristiwa untuk memfilter dan mengirim data ke target.

Menerima acara dari Salesforce menggunakan Amazon AppFlow

Amazon AppFlow merangkum acara dari Salesforce dalam amplop acara. EventBridge Contoh berikut menunjukkan Salesforce acara yang diterima oleh bus acara EventBridge mitra.

```
{
    "version": "0",
    "id": "5c42b99e-e005-43b3-c744-07990c50d2cc",
    "detail-type": "AccountChangeEvent",
    "source": "aws.partner/appflow.test/salesforce.com/364228160620/CustomSF-Source-
Final",
    "account": "000000000",
    "time": "2020-08-20T18:25:51Z",
    "region": "us-west-2",
    "resources": [],
    "detail": {
        "ChangeEventHeader": {
            "commitNumber": 248197218874,
            "commitUser": "0056g000003XW7AAAW",
            "sequenceNumber": 1,
            "entityName": "Account",
            "changeType": "UPDATE",
            "changedFields": [
                "LastModifiedDate",
                "Region__c"
            ],
```

```
            "changeOrigin": "com/salesforce/api/soap/49.0;client=SfdcInternalAPI/",
            "transactionKey": "000035af-b239-0581-9f14-461e4187de11",
            "commitTimestamp": 1597947935000,
            "recordIds": [
                "0016g00000MLhLeAAL"
            ]
        },
        "LastModifiedDate": "2020-08-20T18:25:35.000Z",
        "Region__c": "America"
    }
}
```

Langkah 1: Konfigurasikan Amazon AppFlow untuk digunakan Salesforce sebagai sumber acara mitra

Untuk mengirim acara ke EventBridge, Anda harus terlebih dahulu mengonfigurasi Amazon AppFlow untuk digunakan Salesforce sebagai sumber acara mitra.

1. Di AppFlowkonsol Amazon, pilih Buat alur.
2. Di bagian Detail alur, di Nama alur masukkan nama untuk alur Anda.
3. (Opsional) Masukkan deskripsi untuk alur, lalu pilih Selanjutnya.
4. Di bawah Rincian sumber, pilih Salesforcedari drop-down Nama sumber, lalu pilih Connect untuk membuat koneksi baru.
5. Di kotak Salesforce dialog Connect to, pilih Production atau Sandbox untuk Salesforce lingkungan.
6. Di bidang Nama koneksi, masukkan nama unik untuk sambungan, dan kemudian pilih Lanjutkan.
7. Di kotak Salesforce dialog, lakukan hal berikut:
    a. Masukkan kredensi Salesforce masuk Anda untuk masuk. Salesforce
    b. Pilih Salesforce peristiwa untuk jenis data yang akan diproses Amazon AppFlow .
8. Di menu drop-down Pilih Salesforce acara, pilih jenis acara yang akan dikirim EventBridge.
9. Untuk tujuan, pilih Amazon EventBridge.
10. Pilih Buat sumber peristiwa mitra baru.
11. (Opsional) Tentukan sufiks unik untuk sumber peristiwa mitra.
12. Pilih Hasilkan sumber peristiwa mitra.
13. Pilih bucket Amazon S3 untuk menyimpan file muatan peristiwa yang lebih besar dari 256 KB.

14. Di bagian Pemicu alur, pastikan bahwa Jalankan alur pada peristiwa dipilih. Pengaturan ini memastikan bahwa aliran dijalankan ketika Salesforce peristiwa baru terjadi.
15. Pilih Selanjutnya.
16. Untuk pemetaan bidang, pilih Memetakan semua bidang secara langsung. Atau, Anda dapat memilih bidang yang menarik dari daftar Nama kolom sumber.

    Untuk informasi selengkapnya tentang pemetaan kolom, lihat Memetakan bidang data.
17. Pilih Selanjutnya.
18. (Opsional) Konfigurasikan filter untuk bidang data di Amazon AppFlow.
19. Pilih Selanjutnya.
20. Tinjau pengaturan dan kemudian pilih Buat alur.

Dengan alur yang dikonfigurasi, Amazon AppFlow membuat sumber acara mitra baru yang kemudian perlu Anda kaitkan dengan bus acara mitra di akun Anda.

Langkah 2: Konfigurasikan EventBridge untuk menerima Salesforce acara

Pastikan bahwa AppFlow aliran Amazon yang dipicu dari Salesforce peristiwa dengan EventBridge sebagai tujuan dikonfigurasi sebelum mengikuti petunjuk di bagian ini.

Untuk mengkonfigurasi EventBridge untuk menerima Salesforce acara

1. Buka halaman Sumber acara Partner di EventBridge konsol.
2. Pilih sumber acara Salesforce mitra yang Anda buat di Langkah 1.
3. Pilih Berasosiasi dengan bus peristiwa.
4. Memvalidasi nama bus peristiwa mitra.
5. Pilih Kaitkan.
6. Di AppFlow konsol Amazon, buka alur yang Anda buat dan pilih Aktifkan aliran.
7. Buka halaman Aturan di EventBridge konsol.
8. Pilih Buat aturan.
9. Masukkan nama unik untuk aturan.
10. Pilih Pola peristiwa di bagian Tentukan pola.
11. Untuk Pola pencocokan peristiwa, pilih Pola yang telah ditentukan sebelumnya berdasarkan layanan.
12. UntukPenyedia layananBagian, pilih Semua Peristiwa.

13. UntukPilih bus peristiwa, pilihBus peristiwa kustom atau mitra.
14. Pilih bus acara yang Anda kaitkan dengan sumber acara AppFlow mitra Amazon.
15. Untuk Pilih target, pilih AWS layanan yang akan bertindak saat aturan berjalan. Satu aturan dapat memiliki sampai dengan lima target.
16. Pilih Buat.

Layanan target menerima semua Salesforce peristiwa yang dikonfigurasi untuk akun Anda. Untuk memfilter peristiwa atau mengirim beberapa peristiwa ke target yang berbeda, Anda dapat menggunakan pemfilteran berbasis konten dengan pola acara.

> **Note**
>
> Untuk acara yang lebih besar dari 256KB, Amazon AppFlow tidak mengirim acara lengkap ke. EventBridge Sebagai gantinya, Amazon AppFlow menempatkan acara tersebut ke dalam bucket S3 di akun Anda, lalu mengirimkan acara EventBridge dengan penunjuk ke bucket Amazon S3. Anda dapat menggunakan pointer untuk mendapatkan peristiwa penuh dari bucket.

Mendebug pengiriman EventBridge acara Amazon

Masalah pengiriman acara bisa sulit diidentifikasi, EventBridge menawarkan beberapa cara untuk men-debug dan memulihkan dari kegagalan pengiriman acara.

Topik

- Kebijakan coba lagi peristiwa dan menggunakan antrean surat mati

Kebijakan coba lagi peristiwa dan menggunakan antrean surat mati

Terkadang peristiwa tidak berhasil dikirim ke target yang ditentukan dalam aturan. Ini dapat terjadi ketika, misalnya, sumber daya target tidak tersedia, ketika tidak EventBridge memiliki izin ke sumber daya target, atau karena kondisi jaringan. Ketika suatu peristiwa tidak berhasil dikirim ke target karena kesalahan yang dapat diambil, EventBridge coba lagi mengirim acara tersebut. Anda mengatur panjang waktu mencoba, dan jumlah upaya coba lagi dalam pengaturan Kebijakan coba lagi untuk target. Secara default, EventBridge coba lagi mengirim acara selama 24 jam dan hingga 185 kali dengan mundur dan jitter eksponensial, atau penundaan acak. Jika suatu peristiwa tidak dikirimkan setelah semua upaya percobaan ulang habis, acara dibatalkan dan EventBridge tidak terus memprosesnya. Untuk menghindari kehilangan peristiwa setelah gagal dikirim ke target, Anda dapat mengkonfigurasi antrean surat mati (DLQ) dan mengirim semua peristiwa gagal untuk diproses kemudian.

EventBridge DLQ adalah antrian Amazon SQS standar yang EventBridge digunakan untuk menyimpan peristiwa yang tidak berhasil dikirim ke target. Ketika Anda membuat aturan dan menambahkan target, Anda dapat memilih menggunakan DLQ atau tidak. Bila Anda mengonfigurasi DLQ, Anda dapat mempertahankan setiap peristiwa yang tidak berhasil dikirim. Kemudian Anda dapat menyelesaikan masalah yang mengakibatkan pengiriman peristiwa gagal dan memproses peristiwa di lain waktu.

Kesalahan peristiwa ditangani dengan cara yang berbeda. Beberapa peristiwa dibatalkan atau dikirim ke DLQ tanpa upaya mencoba lagi. Sebagai contoh, untuk kesalahan yang dihasilkan dari izin yang hilang ke target, atau sumber daya target yang tidak lagi ada, semua upaya coba lagi gagal sampai tindakan yang diambil untuk menyelesaikan masalah mendasar. Daripada mencoba lagi, EventBridge kirimkan acara ini langsung ke DLQ, jika Anda memilikinya.

Ketika pengiriman acara gagal, EventBridge memublikasikan peristiwa ke CloudWatch metrik Amazon yang menunjukkan bahwa target `invocation` gagal. Jika Anda menggunakan DLQ, metrik tambahan dikirim ke CloudWatch termasuk dan. `InvocationsSentToDLQ` `InvocationsFailedToBeSentToDLQ`

Setiap pesan di DLQ Anda akan menyertakan atribut kustom berikut:

- `RULE_ARN`
- `TARGET_ARN`
- `ERROR_CODE`

  Berikut ini adalah contoh kode kesalahan yang dapat dikembalikan oleh DLQ:

  - CONNECTION_FAILURE
  - CROSS_ACCOUNT_INGESTION_FAILED
  - CROSS_REGION_INGESTION_FAILED
  - ERROR_FROM_TARGET
  - EVENTS_IN_BATCH_REQUEST_REJECTED
  - EVENTS_IN_BATCH_REQUEST_REJECTED
  - FAILED_TO_ASSUME_ROLE
  - INTERNAL_ERROR
  - INVALID_JSON
  - INVALID_PARAMETER
  - NO_PERMISSIONS
  - NO_RESOURCE
  - RESOURCE_ALREADY_EXISTS
  - RESOURCE_LIMIT_EXCEEDED
  - RESOURCE_MODIFICATION_COLLISION
  - SDK_CLIENT_ERROR
  - THIRD_ACCOUNT_HOP_DETECTED
  - THIRD_REGION_HOP_DETECTED
  - THROTTLING
  - TIMEOUT
  - TRANSIENT_ASSUME_ROLE
  - UNKNOWN
- ERROR_MESSAGE
- EXHAUSTED_RETRY_CONDITION

  Kondisi berikut dapat dikembalikan:

  - MaximumRetryAttempts
  - MaximumEventAgeInSeconds
- RETRY_ATTEMPTS

Video berikut membahas pengaturan DLQ: Menggunakan antrian huruf mati (DLQ)

Topik

- Pertimbangan untuk menggunakan antrean surat mati
- Memberikan izin untuk antrean surat mati
- Bagaimana cara mengirim ulang peristiwa antrean surat mati

Pertimbangan untuk menggunakan antrean surat mati

Pertimbangkan hal berikut saat mengonfigurasi DLQ untuk. EventBridge

- Hanya antrean standar yang didukung. Anda tidak dapat menggunakan antrian FIFO untuk DLQ di. EventBridge
- EventBridge menyertakan metadata peristiwa dan atribut pesan dalam pesan, termasuk: Kode Kesalahan, Pesan Kesalahan, Kondisi Coba Lagi yang Habis, ARN Aturan, Upaya Coba Lagi, dan ARN Target. Anda dapat menggunakan nilai-nilai ini untuk mengidentifikasi peristiwa dan penyebab kegagalan.
- Izin untuk DLQs di akun yang sama:
  - Jika Anda menambahkan target ke aturan menggunakan konsol, dan Anda memilih antrean Amazon SQS di akun yang sama, kebijakan berbasis sumber daya yang memberikan EventBridge akses ke antrian akan dilampirkan ke antrian untuk Anda.
  - Jika Anda menggunakan `PutTargets` pengoperasian EventBridge API untuk menambah atau memperbarui target aturan, dan Anda memilih antrean Amazon SQS di akun yang sama, Anda harus memberikan izin secara manual ke antrian yang dipilih. Untuk mempelajari selengkapnya, lihat Memberikan izin untuk antrean surat mati.
- Izin untuk menggunakan antrian Amazon SQS dari akun yang berbeda. AWS
  - Jika Anda membuat aturan dari konsol tersebut, antrean dari akun lain tidak ditampilkan untuk Anda pilih. Anda harus menyediakan ARN untuk antrean di akun lain, dan kemudian secara manual melampirkan kebijakan berbasis sumber daya untuk memberikan izin untuk antrean. Untuk pelajari selengkapnya, lihat Memberikan izin untuk antrean surat mati.
  - Jika Anda membuat aturan dengan menggunakan API, Anda harus secara manual melampirkan kebijakan berbasis sumber daya untuk antrean SQS di akun lain yang digunakan sebagai antrean surat mati. Untuk pelajari selengkapnya, lihat Memberikan izin untuk antrean surat mati.
- Antrean Amazon SQS yang Anda gunakan harus berada di Wilayah yang sama di mana Anda membuat aturan.

Memberikan izin untuk antrean surat mati

Saat Anda mengonfigurasi DLQ untuk target aturan, EventBridge mengirimkan peristiwa dengan pemanggilan gagal ke antrean Amazon SQS yang dipilih. Agar berhasil mengirimkan acara ke antrian, EventBridge harus memiliki izin untuk melakukannya. Saat Anda mengonfigurasi target untuk aturan dan memilih DLQ menggunakan EventBridge konsol, izin akan ditambahkan secara otomatis. Jika Anda membuat aturan menggunakan API, atau menggunakan antrean yang ada di AWS akun lain, Anda harus secara manual membuat kebijakan berbasis sumber daya yang memberikan izin yang diperlukan dan kemudian melampirkannya ke antrian.

Kebijakan berbasis sumber daya berikut menunjukkan cara memberikan izin yang diperlukan untuk mengirim pesan peristiwa ke antrean EventBridge Amazon SQS. Contoh kebijakan memberikan izin EventBridge layanan untuk menggunakan `SendMessage` operasi untuk mengirim pesan ke antrian bernama "MyEventDLQ”. Antrian harus berada di Wilayah us-west-2 di akun 123456789012. AWS `Condition`Pernyataan tersebut hanya mengizinkan permintaan yang berasal dari aturan bernama "MyTestRule" yang dibuat di Wilayah us-west-2 di akun 123456789012. AWS

```
{
  "Sid": "Dead-letter queue permissions",
  "Effect": "Allow",
  "Principal": {
     "Service": "events.amazonaws.com"
  },
  "Action": "sqs:SendMessage",
  "Resource": "arn:aws:sqs:us-west-2:123456789012:MyEventDLQ",
  "Condition": {
    "ArnEquals": {
      "aws:SourceArn": "arn:aws:events:us-west-2:123456789012:rule/MyTestRule"
    }
  }
}
```

Untuk melampirkan kebijakan ke antrean, gunakan konsol Amazon SQS, buka antrean, lalu pilih Kebijakan akses dan edit kebijakan. Anda juga dapat menggunakan AWS CLI, untuk mempelajari lebih lanjut lihatIzin Amazon SQS.

Bagaimana cara mengirim ulang peristiwa antrean surat mati

Anda dapat memindahkan pesan dari DLQ dengan dua cara:

- Hindari menulis logika konsumen Amazon SQS – Atur DLQ Anda sebagai sumber peristiwa untuk fungsi Lambda untuk menguras DLQ Anda.
- Menulis logika konsumen Amazon SQS — Gunakan Amazon SQS API AWS , SDK, AWS CLI atau untuk menulis logika konsumen khusus untuk polling, pemrosesan, dan penghapusan pesan di DLQ.

Pola EventBridge acara Amazon

Pola peristiwa memiliki struktur yang sama dengan peristiwayang cocok. Aturan menggunakan pola peristiwa untuk memilih peristiwa dan merutekannya ke target. Pola peristiwa cocok dengan peristiwa maupun tidak.

> **Important**
>
> Di EventBridge, dimungkinkan untuk membuat aturan yang dapat menyebabkan higher-than-expected pengisian daya dan pelambatan. Misalnya, Anda dapat secara tidak sengaja membuat aturan yang mengarah ke loop tak terbatas, di mana aturan dijalankan secara rekursif tanpa akhir. Misalkan Anda membuat aturan untuk mendeteksi bahwa ACL telah berubah pada bucket Amazon S3, dan memicu perangkat lunak untuk mengubahnya ke status yang diinginkan. Jika aturan tidak ditulis dengan hati-hati, perubahan berikutnya pada ACL akan mengaktifkan kembali aturan, yang membuat loop tak terbatas.
> Untuk panduan tentang cara menulis aturan dan pola peristiwa yang tepat untuk meminimalkan hasil yang tidak terduga tersebut, lihat ??? dan???.

Video berikut membahas dasar-dasar pola acara: Cara memfilter acara

Topik

- Membuat pola acara
- Contoh peristiwa dan pola peristiwa
- Mencocokkan nilai nol dan string kosong dalam pola acara Amazon EventBridge
- Array dalam pola EventBridge acara Amazon
- Pemfilteran konten dalam pola EventBridge acara Amazon
- Menguji pola acara menggunakan EventBridge Sandbox
- Praktik terbaik saat mendefinisikan pola EventBridge acara Amazon

Acara berikut menunjukkan AWS acara sederhana dari Amazon EC2.

```
{
  "version": "0",
```

```
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "EC2 Instance State-change Notification",
  "source": "aws.ec2",
  "account": "111122223333",
  "time": "2017-12-22T18:43:48Z",
  "region": "us-west-1",
  "resources": [
    "arn:aws:ec2:us-west-1:123456789012:instance/i-1234567890abcdef0"
  ],
  "detail": {
    "instance-id": "i-1234567890abcdef0",
    "state": "terminated"
  }
}
```

Pola peristiwa berikut ini memproses semua peristiwa `instance-termination` Amazon EC2.

```
{
  "source": ["aws.ec2"],
  "detail-type": ["EC2 Instance State-change Notification"],
  "detail": {
    "state": ["terminated"]
  }
}
```

Membuat pola acara

Untuk membuat pola peristiwa, Anda menentukan bidang peristiwa yang ingin Anda cocokkan dengan pola peristiwa. Hanya tetapkan bidang yang Anda gunakan untuk pencocokan. Contoh pola peristiwa sebelumnya hanya memberikan nilai untuk tiga bidang: bidang tingkat atas `"source"` dan`"detail-type"`, dan `"state"` bidang di dalam bidang `"detail"` objek. EventBridge mengabaikan semua bidang lain dalam acara ketika menerapkan aturan.

Untuk pola peristiwa untuk mencocokkan peristiwa, peristiwa harus berisi semua nama bidang yang tercantum dalam pola peristiwa. Nama bidang juga harus muncul dalam peristiwa dengan struktur persarangan yang sama.

Ketika Anda menulis pola peristiwa untuk mencocokkan peristiwa, Anda dapat menggunakan API `TestEventPattern` atau perintah CLI `test-event-pattern` untuk menguji bahwa pola Anda cocok dengan peristiwa yang benar. Untuk informasi lebih lanjut, lihat TestEventPattern.

Mencocokkan nilai acara

Dalam pola peristiwa, nilai yang cocok adalah dalam array JSON, dikelilingi oleh tanda kurung siku (“[”, “]”) sehingga Anda dapat memberikan beberapa nilai. Misalnya, untuk mencocokkan peristiwa dari Amazon EC2 atau AWS Fargate, Anda dapat menggunakan pola berikut, yang cocok dengan peristiwa di mana nilai untuk `"source"` bidang tersebut adalah salah satu atau`"aws.ec2"`. `"aws.fargate"`

```
{
    "source": ["aws.ec2", "aws.fargate"]
}
```

Pertimbangan saat membuat pola acara

Berikut adalah beberapa hal yang perlu dipertimbangkan ketika membangun pola acara Anda:

- EventBridge mengabaikan bidang jika tidak disertakan dalam pola acara. Efeknya adalah ada `"*": "*"` wildcard untuk bidang yang tidak muncul dalam pola acara.
- Nilai yang cocok dengan pola peristiwa mengikuti aturan JSON. Anda dapat menyertakan string yang ditutup dalam tanda kutip ("), angka, dan kata kunci `true`, `false`, dan `null`.
- Untuk string, EventBridge gunakan character-by-character pencocokan yang tepat tanpa case folding atau normalisasi string lainnya.
- Untuk angka, EventBridge gunakan representasi string. Sebagai contoh, 300, 300,0, dan 3,0e2 dianggap tidak sama.
- Jika beberapa pola ditentukan untuk bidang JSON yang sama, EventBridge hanya menggunakan yang terakhir.
- Ketahuilah bahwa ketika EventBridge mengkompilasi pola acara untuk digunakan, ia menggunakan dot (.) sebagai karakter yang bergabung.

  Ini berarti EventBridge akan memperlakukan pola peristiwa berikut sebagai identik:

  ```
  ## has no dots in keys
  { "detail" : { "state": { "status": [ "running" ] } } }

  ## has dots in keys
  { "detail" : { "state.status": [ "running" ] } }
  ```

  Dan bahwa kedua pola acara akan cocok dengan dua peristiwa berikut:

```
## has no dots in keys
{ "detail" : { "state": { "status": "running" } } }

## has dots in keys
{ "detail" : { "state.status": "running"  } }
```

> **Note**
>
> Ini menggambarkan EventBridge perilaku saat ini, dan tidak boleh diandalkan untuk tidak berubah.

- Pola acara yang berisi kolom duplikat tidak valid. Jika pola berisi kolom duplikat, EventBridge hanya mempertimbangkan nilai bidang akhir.

  Misalnya, pola acara berikut akan cocok dengan acara yang sama:

```
## has duplicate keys
{
  "source": ["aws.s3"],
  "source": ["aws.sns"],
  "detail-type": ["AWS API Call via CloudTrail"],
  "detail":  {
      "eventSource": ["s3.amazonaws.com"],
      "eventSource": ["sns.amazonaws.com"]
  }
}

## has unique keys
{
  "source": ["aws.sns"],
  "detail-type": ["AWS API Call via CloudTrail"],
  "detail": { "eventSource": ["sns.amazonaws.com"] }
}
```

  Dan EventBridge memperlakukan dua peristiwa berikut sebagai identik:

```
## has duplicate keys
{
  "source": ["aws.s3"],
  "source": ["aws.sns"],
```

```
  "detail-type": ["AWS API Call via CloudTrail"],
  "detail":  [
    {
      "eventSource": ["s3.amazonaws.com"],
      "eventSource": ["sns.amazonaws.com"]
    }
  ]
}

## has unique keys
{
  "source": ["aws.sns"],
  "detail-type": ["AWS API Call via CloudTrail"],
  "detail": [
    { "eventSource": ["sns.amazonaws.com"] }
  ]
}
```

> **Note**
>
> Ini menggambarkan EventBridge perilaku saat ini, dan tidak boleh diandalkan untuk tidak berubah.

Operasi perbandingan untuk digunakan dalam pola acara

Di bawah ringkasan semua operator perbandingan yang tersedia di EventBridge.

Operator perbandingan hanya bekerja pada node daun, dengan pengecualian `$or` dan`anything-but`.

Perbandingan	Contoh	Sintaks aturan
Dan	Lokasi adalah “New York” dan Hari adalah “Senin”	`"Location": [ "New York" ], "Day": ["Monday"]`
Apa saja-tapi	Negara adalah nilai apa pun selain “menginisialisasi”.	`"state": [ { "anything-but": "initializing" } ]`

Perbandingan	Contoh	Sintaks aturan
Apa saja-tapi (dimulai dengan)	Wilayah tidak ada di AS.	`"Region": [ { "anything-but": {"prefix": "us-" } } ]`
Apa saja-tapi (berakhir dengan)	FileName tidak diakhiri dengan ekstensi.png.	`"FileName": [ { "anything-but": { "suffix": ".png" } } ]`
Apa pun-tapi (abaikan kasus)	Status adalah nilai apa pun selain “inisialisasi” atau variasi casing lainnya, seperti “INISIALISASI”.	`"state": : [{ "anything-but": { "equals-ignore-case": "initializing" }}]}`
Dimulai dengan	Wilayah ada di AS.	`"Region": [ {"prefix": "us-" } ]`
Dimulai dengan (abaikan kasus)	Nama layanan dimulai dengan huruf “eventb”, terlepas dari kasusnya.	`{"service" : [{ "prefix": { "equals-ignore-case": "eventb" }}]}`
Kosong	LastName kosong.	`"LastName": [""]`
Setara	Namanya adalah “Alice”	`"Name": [ "Alice" ]`
Sama dengan (abaikan kasus)	Namanya adalah “Alice”	`"Name": [ { "equals-ignore-case": "alice" } ]`
Berakhir dengan	FileName diakhiri dengan ekstensi. png	`"FileName": [ { "suffix": ".png" } ]`
Berakhir dengan (abaikan kasus)	Nama layanan diakhiri dengan huruf “tbridge”, atau variasi casing lainnya, seperti “TBRIDGE”.	`{"service" : [{ "suffix": { "equals-ignore-case": "tBridge" }}]}`
Ada	ProductName ada	`"ProductName": [ { "exists": true } ]`

Perbandingan	Contoh	Sintaks aturan
Tidak ada	ProductName tidak ada	`"ProductName": [ { "exists": false } ]`
Tidak	Cuaca sama sekali tidak “Hujan”	`"Weather": [ { "anything-but": [ "Raining" ] } ]`
Null	UserID adalah kosong	`"UserID": [ null ]`
Numerik (sama dengan)	Harga 100	`"Price": [ { "numeric": [ "=", 100 ] } ]`
Numerik (rentang)	Harga lebih dari 10, dan kurang dari atau sama dengan 20	`"Price": [ { "numeric": [ ">", 10, "<=", 20 ] } ]`
Atau	PaymentType adalah “Kredit” atau “Debit”	`"PaymentType": [ "Credit", "Debit"]`
Atau (beberapa bidang)	Lokasi adalah “New York”, atau Hari adalah “Senin”.	`"$or": [ { "Location": [ "New York" ] }, { "Day": [ "Monday" ] } ]`
Wildcard	File apa pun dengan ekstensi.png, terletak di dalam folder “dir”	`"FileName": [ { "wildcard": "dir/*.png" } ]`

Contoh peristiwa dan pola peristiwa

Anda dapat menggunakan semua jenis data JSON dan nilai untuk mencocokkan peristiwa. Contoh berikut ini menunjukkan peristiwa dan pola peristiwa yang cocok dengannya.

Pencocokan bidang

Anda dapat mencocokkan pada nilai bidang. Pertimbangkan peristiwa Amazon EC2 Auto Scaling berikut ini.

```
{
  "version": "0",
  "id": "3e3c153a-8339-4e30-8c35-687ebef853fe",
```

```
  "detail-type": "EC2 Instance Launch Successful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2015-11-11T21:31:47Z",
  "region": "us-east-1",
  "resources": [],
  "detail": {
    "eventVersion": "",
    "responseElements": null
  }
}
```

Untuk peristiwa sebelumnya, Anda dapat menggunakan bidang `"responseElements"`untuk mencocokkan.

```
{
  "source": ["aws.autoscaling"],
  "detail-type": ["EC2 Instance Launch Successful"],
  "detail": {
   "responseElements": [null]
  }
}
```

Pencocokan nilai

Pertimbangkan acara Amazon Macie berikut, yang terpotong.

```
{
  "version": "0",
  "id": "0948ba87-d3b8-c6d4-f2da-732a1example",
  "detail-type": "Macie Finding",
  "source": "aws.macie",
  "account": "123456789012",
  "time": "2021-04-29T23:12:15Z",
  "region":"us-east-1",
  "resources": [

  ],
  "detail": {
    "schemaVersion": "1.0",
    "id": "64b917aa-3843-014c-91d8-937ffexample",
    "accountId": "123456789012",
```

```
    "partition": "aws",
    "region": "us-east-1",
    "type": "Policy:IAMUser/S3BucketEncryptionDisabled",
    "title": "Encryption is disabled for the S3 bucket",
    "description": "Encryption is disabled for the Amazon S3 bucket. The data in the
 bucket isn’t encrypted
        using server-side encryption.",
    "severity": {
        "score": 1,
        "description": "Low"
    },
    "createdAt": "2021-04-29T15:46:02Z",
    "updatedAt": "2021-04-29T23:12:15Z",
    "count": 2,
.
.
.
```

Pola acara berikut cocok dengan setiap peristiwa yang memiliki skor keparahan 1 dan hitungan 2.

```
{
  "source": ["aws.macie"],
  "detail-type": ["Macie Finding"],
  "detail": {
    "severity": {
      "score": [1]
    },
    "count":[2]
  }
}
```

Mencocokkan nilai nol dan string kosong dalam pola acara Amazon EventBridge

> **Important**
>
> Di EventBridge, dimungkinkan untuk membuat aturan yang dapat menyebabkan higher-than-expected pengisian daya dan pelambatan. Misalnya, Anda dapat secara tidak sengaja membuat aturan yang mengarah ke loop tak terbatas, di mana aturan dijalankan secara rekursif tanpa akhir. Misalkan Anda membuat aturan untuk mendeteksi bahwa ACL telah berubah pada bucket Amazon S3, dan memicu perangkat lunak untuk mengubahnya ke status yang diinginkan. Jika aturan tidak ditulis dengan hati-hati, perubahan berikutnya pada ACL akan mengaktifkan kembali aturan, yang membuat loop tak terbatas.
> Untuk panduan tentang cara menulis aturan dan pola peristiwa yang tepat untuk meminimalkan hasil yang tidak terduga tersebut, lihat ??? dan???.

Anda dapat membuat pola peristiwa yang cocok dengan bidang di peristiwa yang memiliki nilai null atau string kosong. Pertimbangkan peristiwa contoh berikut ini.

Lihat praktik terbaik untuk menghindari biaya dan pelambatan yang lebih tinggi dari yang diharapkan

```
{
  "version": "0",
  "id": "3e3c153a-8339-4e30-8c35-687ebef853fe",
  "detail-type": "EC2 Instance Launch Successful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2015-11-11T21:31:47Z",
  "region": "us-east-1",
  "resources": [
   ],
  "detail": {
    "eventVersion": "",
    "responseElements": null
   }
}
```

Untuk mencocokkan peristiwa di mana nilai `eventVersion` adalah string kosong, gunakan pola peristiwa berikut ini, yang cocok dengan peristiwa sebelumnya.

```
{
  "detail": {
     "eventVersion": [""]
  }
}
```

Untuk mencocokkan peristiwa di mana nilai `responseElements` adalah null, gunakan pola peristiwa berikut ini, yang cocok dengan peristiwa sebelumnya.

```
{
  "detail": {
     "responseElements": [null]
  }
}
```

> **Note**
>
> Nilai null dan string kosong tidak dapat dipertukarkan dalam pencocokan pola. Pola peristiwa yang cocok dengan string kosong tidak cocok dengan nilai `null`.

Array dalam pola EventBridge acara Amazon

Nilai dari setiap bidang dalam pola peristiwa adalah array yang berisi satu atau lebih nilai. Pola peristiwa yang cocok dengan peristiwa jika salah satu nilai dalam array cocok dengan nilai dalam peristiwa. Jika nilai dalam peristiwa adalah array, maka pola peristiwa cocok jika persimpangan array pola peristiwa dan array peristiwa adalah bukang kosong.

> **Important**
>
> Di EventBridge, dimungkinkan untuk membuat aturan yang dapat menyebabkan higher-than-expected pengisian daya dan pelambatan. Misalnya, Anda dapat secara tidak sengaja membuat aturan yang mengarah ke loop tak terbatas, di mana aturan dijalankan secara rekursif tanpa akhir. Misalkan Anda membuat aturan untuk mendeteksi bahwa ACL telah berubah pada bucket Amazon S3, dan memicu perangkat lunak untuk mengubahnya ke status yang diinginkan. Jika aturan tidak ditulis dengan hati-hati, perubahan berikutnya pada ACL akan mengaktifkan kembali aturan, yang membuat loop tak terbatas.
> Untuk panduan tentang cara menulis aturan dan pola peristiwa yang tepat untuk meminimalkan hasil yang tidak terduga tersebut, lihat ??? dan???.

Sebagai contoh, pertimbangkan pola peristiwa yang mencakup bidang berikut ini.

```
"resources": [
   "arn:aws:ec2:us-east-1:123456789012:instance/i-b188560f",
   "arn:aws:ec2:us-east-1:111122223333:instance/i-b188560f",
   "arn:aws:ec2:us-east-1:444455556666:instance/i-b188560f",
]
```

Pola peristiwa sebelumnya cocok dengan peristiwa yang mencakup bidang berikut ini karena item pertama dalam array pola peristiwa cocok dengan item kedua dalam array peristiwa.

```
"resources": [
   "arn:aws:autoscaling:us-east-1:123456789012:autoScalingGroup:eb56d16b-bbf0-401d-
b893-d5978ed4a025:autoScalingGroupName/ASGTerminate",
   "arn:aws:ec2:us-east-1:123456789012:instance/i-b188560f"
]
```

Pemfilteran konten dalam pola EventBridge acara Amazon

Amazon EventBridge mendukung penyaringan konten deklaratif menggunakan pola peristiwa. Dengan penyaringan konten, Anda dapat menulis pola peristiwa kompleks yang hanya cocok dengan peristiwa menurut kondisi yang sangat spesifik. Misalnya, Anda dapat membuat pola acara yang cocok dengan acara saat:

- Bidang acara berada dalam rentang numerik tertentu.
- Acara ini berasal dari alamat IP tertentu.
- Bidang tertentu tidak ada di acara JSON.

> **Important**
>
> Di EventBridge, dimungkinkan untuk membuat aturan yang dapat menyebabkan higher-than-expected pengisian daya dan pelambatan. Misalnya, Anda dapat secara tidak sengaja membuat aturan yang mengarah ke loop tak terbatas, di mana aturan dijalankan secara rekursif tanpa akhir. Misalkan Anda membuat aturan untuk mendeteksi bahwa ACL telah berubah pada bucket Amazon S3, dan memicu perangkat lunak untuk mengubahnya ke status yang diinginkan. Jika aturan tidak ditulis dengan hati-hati, perubahan berikutnya pada ACL akan mengaktifkan kembali aturan, yang membuat loop tak terbatas.
> Untuk panduan tentang cara menulis aturan dan pola peristiwa yang tepat untuk meminimalkan hasil yang tidak terduga tersebut, lihat ??? dan???.

Jenis filter

- Pencocokan prefiks
- Pencocokan akhiran
- Apa pun yang cocok
- Pencocokan numerik
- Pencocokan alamat IP
- Pencocokan yang ada
- quals-ignore-case Pencocokan E
- Pencocokan menggunakan wildcard
- Contoh kompleks dengan beberapa pencocokan

- Contoh kompleks dengan $or pencocokan

Pencocokan prefiks

Anda dapat mencocokkan peristiwa bergantung pada prefiks nilai dalam sumber peristiwa. Anda dapat menggunakan pencocokan awalan untuk nilai string.

Sebagai contoh, pola peristiwa berikut ini akan cocok dengan peristiwa di mana bidang `"time"` dimulai dengan `"2017-10-02"` seperti `"time": "2017-10-02T18:43:48Z"`.

```
{
  "time": [ { "prefix": "2017-10-02" } ]
}
```

Pencocokan awalan sambil mengabaikan kasus

Anda juga dapat mencocokkan nilai awalan terlepas dari casing karakter yang dimulai dengan nilai, menggunakan `equals-ignore-case` bersama dengan `prefix`.

Misalnya, pola peristiwa berikut akan cocok dengan peristiwa di mana `service` bidang dimulai dengan string karakter`EventB`, tetapi juga`EVENTB`,`eventb`, atau huruf besar lainnya dari karakter tersebut.

```
{
  "detail": {"service" : [{ "prefix": { "equals-ignore-case": "EventB" }}]}
}
```

Pencocokan akhiran

Anda dapat mencocokkan acara tergantung pada akhiran nilai di sumber acara. Anda dapat menggunakan pencocokan akhiran untuk nilai string.

Misalnya, pola acara berikut akan cocok dengan acara apa pun di mana `"FileName"` bidang diakhiri dengan ekstensi `.png` file.

```
{
  "FileName": [ { "suffix": ".png" } ]
}
```

Pencocokan akhiran sambil mengabaikan kasus

Anda juga dapat mencocokkan nilai akhiran terlepas dari casing karakter yang diakhiri dengan nilai, menggunakan `equals-ignore-case` bersama dengan `suffix.`

Misalnya, pola peristiwa berikut akan cocok dengan peristiwa di mana `FileName` bidang diakhiri dengan string karakter `.png`, tetapi juga `.PNG` atau huruf besar lainnya dari karakter tersebut.

```
{
  "detail": {"FileName" : [{ "suffix": { "equals-ignore-case": ".png" }}]}
}
```

Apa pun yang cocok

Apa pun yang cocok akan cocok dengan apa pun kecuali yang disediakan dalam aturan.

Anda dapat menggunakan apa pun yang cocok dengan nilai string dan nilai numerik, termasuk daftar yang hanya berisi string, atau hanya angka.

Pola peristiwa berikut ini menunjukkan apa pun yang cocok dengan string dan angka.

```
{
  "detail": {
    "state": [ { "anything-but": "initializing" } ]
  }
}

{
  "detail": {
    "x-limit": [ { "anything-but": 123 } ]
  }
}
```

Pola peristiwa berikut ini menunjukkan apa pun yang cocok dengan daftar string.

```
{
  "detail": {
    "state": [ { "anything-but": [ "stopped", "overloaded" ] } ]
  }
}
```

Pola peristiwa berikut ini menunjukkan apa pun yang cocok dengan daftar angka.

```
{
  "detail": {
    "x-limit": [ { "anything-but": [ 100, 200, 300 ] } ]
  }
}
```

Apa pun-kecuali cocok sambil mengabaikan kasus

Anda juga dapat menggunakan `equals-ignore-case` bersama dengan`anything-but`, untuk mencocokkan nilai string terlepas dari casing karakter.

Pola acara berikut cocok dengan `state` bidang yang berisi string “inisialisasi”, tetapi juga “INISIALISASI”, “Inisialisasi”, atau huruf besar lainnya dari karakter tersebut.

```
{
  "detail": {"state" : [{ "anything-but": { "equals-ignore-case": "initializing" }}]}
}
```

Anda dapat menggunakan `equals-ignore-case` bersama dengan `anything-but` untuk mencocokkan dengan daftar nilai juga:

```
{
  "detail": {"state" : [{ "anything-but": { "equals-ignore-case": ["initializing",
 "stopped"] }}]}
}
```

Apa pun-kecuali cocok pada awalan

Pola peristiwa berikut ini menunjukkan apa pun yang cocok yang cocok dengan peristiwa apa pun yang tidak memiliki prefiks `"init"` dalam bidang `"state"`.

> **Note**
>
> Apa pun-kecuali pencocokan hanya berfungsi dengan awalan tunggal, bukan daftar.

```
{
```

```
  "detail": {
    "state": [ { "anything-but": { "prefix": "init" } } ]
  }
}
```

Apa pun-kecuali cocok pada sufiks

Anda dapat menggunakan `suffix` bersama dengan `anything-but` untuk mencocokkan nilai string terlepas dari casing karakter.

> **Note**
>
> Apa pun-kecuali pencocokan hanya berfungsi dengan satu sufiks, bukan daftar.

Pola acara berikut cocok dengan nilai apapun untuk `FileName` bidang yang diakhiri dengan `.txt`.

```
{
  "detail": {
    "FileName": [ { "anything-but": { "suffix": ".txt" } } ]
  }
}
```

Pencocokan numerik

Pencocokan numerik bekerja dengan nilai yang merupakan angka JSON. Ini terbatas pada nilai antara -5.0e9 dan +5.0e9 inklusif, dengan 15 digit presisi, atau enam digit di sebelah kanan titik desimal.

Berikut ini menunjukkan pencocokan numerik untuk pola peristiwa yang hanya cocok dengan peristiwa yang betul untuk semua bidang.

```
{
  "detail": {
    "c-count": [ { "numeric": [ ">", 0, "<=", 5 ] } ],
    "d-count": [ { "numeric": [ "<", 10 ] } ],
    "x-limit": [ { "numeric": [ "=", 3.018e2 ] } ]
  }
}
```

Pencocokan alamat IP

Anda dapat menggunakan alamat IP yang cocok untuk alamat IPv4 dan IPv6. Pola peristiwa berikut menunjukkan alamat IP yang cocok dengan alamat IP yang dimulai dengan 10.0.0 dan diakhiri dengan angka antara 0 dan 255.

```
{
  "detail": {
    "sourceIPAddress": [ { "cidr": "10.0.0.0/24" } ]
  }
}
```

Pencocokan yang ada

Pencocokan yang ada bekerja pada ada tidaknya bidang di JSON peristiwa.

Pencocokan yang ada hanya bekerja pada simpul daun. Pencocokan yang ada tidak berfungsi pada simpul intermediet.

Pola acara berikut cocok dengan setiap peristiwa yang memiliki `detail.state` bidang.

```
{
  "detail": {
    "state": [ { "exists": true  } ]
  }
}
```

Pola peristiwa sebelumnya cocok dengan peristiwa berikut ini.

```
{
  "version": "0",
  "id": "7bf73129-1428-4cd3-a780-95db273d1602",
  "detail-type": "EC2 Instance State-change Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2015-11-11T21:29:54Z",
  "region": "us-east-1",
  "resources": ["arn:aws:ec2:us-east-1:123456789012:instance/i-abcd1111"],
  "detail": {
    "instance-id": "i-abcd1111",
    "state": "pending"
  }
```

```
}
```

Pola peristiwa sebelumnya TIDAK cocok dengan peristiwa berikut karena tidak memiliki `detail.state` bidang.

```
{
  "detail-type": [ "EC2 Instance State-change Notification" ],
  "resources": [ "arn:aws:ec2:us-east-1:123456789012:instance/i-02ebd4584a2ebd341" ],
  "detail": {
    "c-count" : {
       "c1" : 100
    }
  }
}
```

quals-ignore-case Pencocokan E

quals-ignore-casePencocokan E bekerja pada nilai string terlepas dari kasusnya.

Pola peristiwa berikut cocok dengan peristiwa apa pun yang memiliki `detail-type` bidang yang cocok dengan string yang ditentukan, apa pun kasusnya.

```
{
  "detail-type": [ { "equals-ignore-case": "ec2 instance state-change notification" } ]
}
```

Pola peristiwa sebelumnya cocok dengan peristiwa berikut ini.

```
{
  "detail-type": [ "EC2 Instance State-change Notification" ],
  "resources": [ "arn:aws:ec2:us-east-1:123456789012:instance/i-02ebd4584a2ebd341" ],
  "detail": {
    "c-count" : {
       "c1" : 100
    }
  }
}
```

Pencocokan menggunakan wildcard

Anda dapat menggunakan karakter wildcard (*) untuk mencocokkan nilai string dalam pola acara.

> **Note**
>
> Saat ini karakter wildcard didukung hanya dalam aturan bus acara.

Pertimbangan saat menggunakan wildcard dalam pola acara Anda:

- Anda dapat menentukan sejumlah karakter wildcard dalam nilai string tertentu; Namun, karakter wildcard berturut-turut tidak didukung.
- EventBridge mendukung penggunaan karakter garis miring terbalik (\) untuk menentukan karakter * dan\ literal dalam filter wildcard:
  - String * mewakili karakter * literal
  - String \\ mewakili karakter\ literal

  Menggunakan garis miring terbalik untuk melarikan diri dari karakter lain tidak didukung.

Wildcard dan kompleksitas pola acara

Ada batasan seberapa rumit aturan menggunakan wildcard. Jika aturan terlalu rumit, EventBridge mengembalikan `InvalidEventPatternException` ketika mencoba untuk membuat aturan. Jika aturan Anda menghasilkan kesalahan seperti itu, pertimbangkan untuk menggunakan panduan di bawah ini untuk mengurangi kompleksitas pola acara:

- Mengurangi jumlah karakter wildcard yang digunakan

  Hanya gunakan karakter wildcard di mana Anda benar-benar perlu mencocokkan dengan beberapa nilai yang mungkin. Misalnya, pertimbangkan pola acara berikut, di mana Anda ingin mencocokkan dengan bus acara di Wilayah yang sama:

```
{
"EventBusArn": [ { "wildcard": "*:*:*:*:*:event-bus/*" } ]
}
```

  Dalam kasus di atas, banyak bagian dari ARN akan langsung didasarkan pada Wilayah di mana bus acara Anda berada. Jadi jika Anda menggunakan `us-east-1` Region, pola yang kurang kompleks yang masih cocok dengan nilai yang diinginkan mungkin adalah contoh berikut:

```
{
```

```
"EventBusArn": [ { "wildcard": "arn:aws:events:us-east-1:*:event-bus/*" } ]
}
```

- Kurangi urutan karakter berulang yang terjadi setelah karakter wildcard

  Memiliki urutan karakter yang sama muncul beberapa kali setelah penggunaan wildcard meningkatkan kompleksitas pemrosesan pola acara. Sesuaikan kembali pola acara Anda untuk meminimalkan urutan berulang. Misalnya, perhatikan contoh berikut, yang cocok pada file nama `doc.txt` file untuk setiap pengguna:

  ```
  {
  "FileName": [ { "wildcard": "/Users/*/dir/dir/dir/dir/dir/doc.txt" } ]
  }
  ```

  Jika Anda tahu bahwa `doc.txt` file hanya akan terjadi di jalur yang ditentukan, Anda dapat mengurangi urutan karakter berulang dengan cara ini:

  ```
  {
  "FileName": [ { "wildcard": "/Users/*/doc.txt" } ]
  }
  ```

Contoh kompleks dengan beberapa pencocokan

Anda dapat menggabungkan beberapa aturan pencocokan ke dalam pola peristiwa yang lebih kompleks. Sebagai contoh, pola peristiwa berikut ini menggabungkan `anything-but` dan `numeric`.

```
{
  "time": [ { "prefix": "2017-10-02" } ],
  "detail": {
    "state": [ { "anything-but": "initializing" } ],
    "c-count": [ { "numeric": [ ">", 0, "<=", 5 ] } ],
    "d-count": [ { "numeric": [ "<", 10 ] } ],
    "x-limit": [ { "anything-but": [ 100, 200, 300 ] } ]
  }
}
```

> **Note**
>
> Saat membangun pola acara, jika Anda menyertakan kunci lebih dari sekali, referensi terakhir akan menjadi referensi yang digunakan untuk mengevaluasi peristiwa. Misalnya, untuk pola berikut:
>
> ```
> {
>   "detail": {
>     "location": [ { "prefix": "us-" } ],
>     "location": [ { "anything-but": "us-east" } ]
>   }
> }
> ```
>
> hanya `{ "anything-but": "us-east" }` akan diperhitungkan saat mengevaluasi. `location`

Contoh kompleks dengan **`$or`** pencocokan

Anda juga dapat membuat pola peristiwa kompleks yang memeriksa untuk melihat apakah ada nilai bidang yang cocok, di beberapa bidang. Gunakan `$or` untuk membuat pola acara yang cocok jika salah satu nilai untuk beberapa bidang cocok.

Perhatikan bahwa Anda dapat menyertakan jenis filter lain, seperti pencocokan numerik dan array, dalam pencocokan pola Anda untuk masing-masing bidang dalam konstruksi Anda`$or`.

Pola acara berikut cocok jika salah satu dari kondisi berikut terpenuhi:

- `c-count`Bidang lebih besar dari 0 atau kurang dari atau sama dengan 5.
- `d-count`Bidang kurang dari 10.
- `x-limit`Bidang sama dengan 3.018e2.

```
{
  "detail": {
    "$or": [
      { "c-count": [ { "numeric": [ ">", 0, "<=", 5 ] } ] },
      { "d-count": [ { "numeric": [ "<", 10 ] } ] },
      { "x-limit": [ { "numeric": [ "=", 3.018e2 ] } ] }
```

```
    ]
  }
}
```

> **Note**
>
> API yang menerima pola peristiwa (sepertiPutRule,CreateArchive,UpdateArchive, danTestEventPattern) akan menampilkan InvalidEventPatternException jika penggunaan $or menghasilkan lebih dari 1000 kombinasi aturan.
> Untuk menentukan jumlah kombinasi aturan dalam pola peristiwa, kalikan jumlah total argumen dari setiap $or array dalam pola peristiwa. Misalnya, pola di atas berisi $or array tunggal dengan tiga argumen, sehingga jumlah total kombinasi aturan juga tiga. Jika Anda menambahkan $or array lain dengan dua argumen, kombinasi aturan total akan menjadi enam.

Menguji pola acara menggunakan EventBridge Sandbox

Aturan menggunakan pola peristiwa untuk memilih peristiwa dan merutekannya ke target. Pola peristiwa memiliki struktur yang sama dengan peristiwa yang dicocokkan. Pola peristiwa bisa cocok atau tidak dengan peristiwa.

Mendefinisikan pola acara biasanya merupakan bagian dari proses yang lebih besar untuk membuat aturan baru atau mengedit yang sudah ada. Namun EventBridge, dengan menggunakan Sandbox di, Anda dapat dengan cepat menentukan pola peristiwa dan menggunakan contoh peristiwa untuk mengonfirmasi pola cocok dengan peristiwa yang diinginkan, tanpa harus membuat atau mengedit aturan. Setelah pola acara Anda diuji, EventBridge beri Anda opsi untuk membuat aturan baru menggunakan pola peristiwa itu langsung dari kotak pasir.

Untuk informasi selengkapnya tentang pola acara, lihat???.

> **Important**
>
> Di EventBridge, dimungkinkan untuk membuat aturan yang dapat menyebabkan higher-than-expected pengisian daya dan pelambatan. Misalnya, Anda dapat secara tidak sengaja membuat aturan yang mengarah ke loop tak terbatas, di mana aturan dijalankan secara rekursif tanpa akhir. Misalkan Anda membuat aturan untuk mendeteksi bahwa ACL telah berubah pada bucket Amazon S3, dan memicu perangkat lunak untuk mengubahnya ke

status yang diinginkan. Jika aturan tidak ditulis dengan hati-hati, perubahan berikutnya pada ACL akan mengaktifkan kembali aturan, yang membuat loop tak terbatas.
Untuk panduan tentang cara menulis aturan dan pola peristiwa yang tepat untuk meminimalkan hasil yang tidak terduga tersebut, lihat ??? dan???.

Untuk menguji pola acara menggunakan kotak EventBridge pasir

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Sumber daya pengembang, lalu pilih Sandbox, dan pada halaman Sandbox pilih tab Pola acara.
3. Untuk sumber Acara, pilih AWS acara atau acara EventBridge mitra.
4. Di bagian Contoh peristiwa, pilih jenis peristiwa Contoh yang ingin Anda uji pola acara Anda.

   Jenis acara sampel berikut tersedia:

   - AWS event - Pilih dari peristiwa yang dipancarkan dari didukung. Layanan AWS
   - EventBridge acara mitra - Pilih dari acara yang dipancarkan dari layanan pihak ketiga yang mendukung EventBridge, seperti Salesforce.
   - Masukkan sendiri - Masukkan acara Anda sendiri dalam teks JSON.

     Anda juga dapat menggunakan acara AWS atau mitra sebagai titik awal untuk membuat acara kustom Anda sendiri.

     1. Pilih AWS acara atau acara EventBridge mitra.
     2. Gunakan dropdown Contoh peristiwa untuk memilih acara yang ingin Anda gunakan sebagai titik awal untuk acara kustom Anda.

        EventBridge menampilkan acara sampel.
     3. Pilih Salin.
     4. Pilih Masukkan milik saya untuk jenis Acara.
     5. Hapus struktur peristiwa sampel di panel pengeditan JSON, dan tempel acara AWS atau mitra di tempatnya.
     6. Edit acara JSON untuk membuat acara sampel Anda sendiri.
5. Pilih metode Creation. Anda dapat membuat pola acara dari EventBridge skema atau template, atau Anda dapat membuat pola acara khusus.

Existing schema

Untuk menggunakan EventBridge skema yang ada untuk membuat pola acara, lakukan hal berikut:

1. Di bagian Metode pembuatan, untuk Metode, pilih Gunakan skema.
2. Di bagian Pola acara, untuk jenis Skema, pilih Pilih skema dari registri Skema.
3. Untuk registri Schema, pilih kotak dropdown dan masukkan nama registri skema, seperti. `aws.events` Anda juga dapat memilih opsi dari daftar dropdown yang muncul.
4. Untuk Skema, pilih kotak dropdown dan masukkan nama skema yang akan digunakan. Misalnya, `aws.s3@0bjectDeleted`. Anda juga dapat memilih opsi dari daftar dropdown yang muncul.
5. Di bagian Model, pilih tombol Edit di sebelah atribut apa pun untuk membuka propertinya. Atur bidang Relationship dan Value sesuai kebutuhan, lalu pilih Set untuk menyimpan atribut.

   > **Note**
   >
   > Untuk informasi tentang definisi atribut, pilih ikon Info di sebelah nama atribut.
   > Untuk referensi tentang cara mengatur properti atribut dalam acara Anda, buka bagian Catatan pada kotak dialog properti atribut.
   > Untuk menghapus properti atribut, pilih tombol Edit untuk atribut tersebut, lalu pilih Hapus.

6. Pilih Hasilkan pola acara di JSON untuk menghasilkan dan memvalidasi pola acara Anda sebagai teks JSON.
7. Untuk menguji peristiwa sampel terhadap pola pengujian Anda, pilih Pola uji.

   EventBridge menampilkan kotak pesan yang menyatakan apakah acara sampel Anda cocok dengan pola acara.

   Anda juga dapat memilih salah satu opsi berikut:

   - Salin — Salin pola acara ke clipboard perangkat Anda.
   - Prettify - Membuat teks JSON lebih mudah dibaca dengan menambahkan jeda baris, tab, dan spasi.

Custom schema

Untuk menulis skema kustom dan mengubahnya menjadi pola acara, lakukan hal berikut:

1. Di bagian Metode pembuatan, untuk Metode, pilih Gunakan skema.
2. Di bagian Pola acara, untuk jenis Skema, pilih Masukkan skema.
3. Masukkan skema Anda ke dalam kotak teks. Anda harus memformat skema sebagai teks JSON yang valid.
4. Di bagian Model, pilih tombol Edit di sebelah atribut apa pun untuk membuka propertinya. Atur bidang Relationship dan Value sesuai kebutuhan, lalu pilih Set untuk menyimpan atribut.

   > **Note**
   >
   > Untuk informasi tentang definisi atribut, pilih ikon Info di sebelah nama atribut.
   > Untuk referensi tentang cara mengatur properti atribut dalam acara Anda, buka bagian Catatan pada kotak dialog properti atribut.
   > Untuk menghapus properti atribut, pilih tombol Edit untuk atribut tersebut, lalu pilih Hapus.

5. Pilih Hasilkan pola acara di JSON untuk menghasilkan dan memvalidasi pola acara Anda sebagai teks JSON.
6. Untuk menguji peristiwa sampel terhadap pola pengujian Anda, pilih Pola uji.

   EventBridge menampilkan kotak pesan yang menyatakan apakah acara sampel Anda cocok dengan pola acara.

   Anda juga dapat memilih salah satu opsi berikut:

   - Salin — Salin pola acara ke clipboard perangkat Anda.
   - Prettify - Membuat teks JSON lebih mudah dibaca dengan menambahkan jeda baris, tab, dan spasi.

Event pattern

Untuk menulis pola acara khusus dalam format JSON, lakukan hal berikut:

1. Di bagian Metode pembuatan, untuk Metode, pilih Pola kustom (editor JSON).
2. Untuk pola Acara, masukkan pola acara kustom Anda dalam teks berformat JSON.
3. Untuk menguji peristiwa sampel terhadap pola pengujian Anda, pilih Pola uji.

   EventBridge menampilkan kotak pesan yang menyatakan apakah acara sampel Anda cocok dengan pola acara.

   Anda juga dapat memilih salah satu opsi berikut:

   - Salin — Salin pola acara ke clipboard perangkat Anda.
   - Prettify - Membuat teks JSON lebih mudah dibaca dengan menambahkan jeda baris, tab, dan spasi.
   - Bentuk pola acara - Membuka pola acara di Pattern Builder. Jika pola tidak dapat dirender di Pattern Builder apa adanya, EventBridge memperingatkan Anda sebelum membuka Pattern Builder.

6. (Opsional) Untuk membuat aturan dengan pola acara ini, dan menetapkan aturan ke bus acara tertentu, pilih Buat aturan dengan pola.

   EventBridge membawa Anda ke Langkah 1 dari Buat aturan, yang dapat Anda gunakan untuk membuat aturan dan menetapkannya ke bus acara pilihan Anda.

   Perhatikan bahwa Langkah 2 - Membangun pola acara berisi informasi pola peristiwa yang telah Anda tentukan, dan yang dapat Anda terima atau perbarui.

   Untuk informasi lebih lanjut tentang cara membuat aturan, lihat???.

Praktik terbaik saat mendefinisikan pola EventBridge acara Amazon

Di bawah ini adalah beberapa praktik terbaik untuk dipertimbangkan saat menentukan pola acara dalam aturan bus acara Anda.

Hindari menulis loop tak terbatas

Di EventBridge, dimungkinkan untuk membuat aturan yang mengarah ke loop tak terbatas, di mana aturan ditembakkan berulang kali. Sebagai contoh, aturan mungkin mendeteksi bahwa ACL telah berubah di bucket S3, lalu memicu perangkat lunak untuk mengubahnya ke keadaan

yang diinginkan. Jika aturan tidak ditulis dengan hati-hati, perubahan berikutnya pada ACL akan mengaktifkan kembali aturan, yang membuat loop tak terbatas.

Untuk mencegah masalah ini, tulis pola acara agar aturan Anda setepat mungkin, sehingga hanya cocok dengan peristiwa yang sebenarnya ingin Anda kirim ke target. Dalam contoh di atas, Anda akan membuat pola acara untuk mencocokkan peristiwa sehingga tindakan yang dipicu tidak mengaktifkan kembali aturan yang sama. Misalnya, buat pola acara dalam aturan Anda yang akan cocok dengan peristiwa hanya jika ACL ditemukan dalam keadaan buruk, bukan setelah perubahan apa pun. Lihat informasi yang lebih lengkap di ??? dan ???.

Loop tak terbatas dapat dengan cepat mengakibatkan biaya yang lebih tinggi dari yang diperkirakan. Ini juga dapat menyebabkan pelambatan dan pengiriman acara tertunda. Anda dapat memantau batas atas tingkat doa Anda untuk diperingatkan tentang lonjakan volume yang tidak terduga.

Gunakan penganggaran untuk mengingatkan Anda ketika biaya melebihi batas yang Anda tentukan. Untuk informasi lebih lanjut, lihat Mengelola Biaya Anda dengan Anggaran.

Buat pola acara setepat mungkin

Semakin tepat pola acara Anda, semakin besar kemungkinannya hanya cocok dengan peristiwa yang benar-benar Anda inginkan, dan menghindari kecocokan tak terduga saat acara baru ditambahkan ke sumber acara, atau acara yang ada diperbarui untuk menyertakan properti baru.

Pola acara dapat mencakup filter yang cocok pada:

- Metadata acara tentang acara tersebut, sepertisource,. detail-type account, atauregion.
- Data peristiwa, ini adalah, bidang di dalam detail objek.
- Konten acara, atau nilai aktual dari bidang di dalam detail objek.

Sebagian besar pola sederhana, seperti hanya menentukan source dan detail-type filter. Namun, EventBridge pola mencakup fleksibilitas untuk memfilter pada kunci atau nilai acara apa pun. Selain itu, Anda dapat menerapkan filter konten seperti prefix dan suffix filter untuk meningkatkan ketepatan pola Anda. Untuk informasi selengkapnya, lihat ???.

Tentukan sumber acara dan jenis detail sebagai filter

Anda dapat mengurangi menghasilkan loop tak terbatas dan mencocokkan peristiwa yang tidak diinginkan dengan membuat pola acara Anda lebih tepat menggunakan bidang source dan detail-type metadata.

Saat Anda perlu mencocokkan nilai tertentu dalam dua bidang atau lebih, gunakan operator `$or` perbandingan, daripada mencantumkan semua nilai yang mungkin dalam satu larik nilai.

Untuk acara yang disampaikan melalui AWS CloudTrail, kami sarankan Anda menggunakan `eventName` bidang sebagai filter.

Contoh pola peristiwa berikut cocok `CreateQueue` atau `SetQueueAttributes` dari layanan Amazon Simple Queue Service, atau `CreateKey` atau `DisableKeyRotation` peristiwa dari AWS Key Management Service layanan.

```
{
  "detail-type": ["AWS API Call via CloudTrail"],
  "$or": [{
      "source": [
        "aws.sqs"
        ],
      "detail": {
        "eventName": [
          "CreateQueue",
          "SetQueueAttributes"
        ]
      }
    },
    {
      "source": [
        "aws.kms"
        ],
      "detail": {
        "eventName": [
          "CreateKey",
          "DisableKeyRotation"
        ]
      }
    }
  ]
}
```

Tentukan akun dan wilayah sebagai filter

`account`Menyertakan dan `region` bidang dalam pola acara Anda membantu membatasi pencocokan acara lintas akun atau lintas wilayah.

Tentukan filter konten

Pemfilteran berbasis konten dapat membantu meningkatkan presisi pola acara, sambil tetap menjaga panjang pola acara seminimal mungkin. Misalnya, pencocokan berdasarkan rentang numerik dapat membantu alih-alih mencantumkan semua nilai numerik yang mungkin.

Untuk informasi selengkapnya, lihat ???.

Cakupan pola acara Anda untuk memperhitungkan pembaruan sumber acara

Saat membuat pola acara, Anda harus mempertimbangkan bahwa skema acara dan domain acara dapat berkembang dan berkembang seiring waktu. Di sini sekali lagi, membuat pola acara Anda seakurat mungkin membantu Anda membatasi kecocokan tak terduga jika sumber acara berubah atau berkembang.

Misalnya, misalkan Anda cocok dengan peristiwa dari layanan mikro baru yang menerbitkan peristiwa terkait pembayaran. Awalnya, layanan menggunakan domain`acme.payments`, dan menerbitkan satu acara,`Payment accepted`:

```
{
  "detail-type": "Payment accepted",
  "source": "acme.payments",
  "detail": {
    "type": "credit",
    "amount": "100",
    "date": "2023-06-10",
    "currency": "USD"
    }
  }
}
```

Pada titik ini, Anda dapat membuat pola acara sederhana yang cocok dengan peristiwa yang diterima Pembayaran:

```
{ “source” : “acme.payments” }
```

Namun, misalkan layanan kemudian memperkenalkan acara baru untuk pembayaran yang ditolak:

```
{
```

```
  "detail-type": "Payment rejected",
  "source": "acme.payments",
  "detail": {
  }
}
```

Dalam hal ini, pola acara sederhana yang Anda buat sekarang akan cocok dengan keduanya `Payment accepted` dan `Payment rejected` peristiwa. EventBridge merutekan kedua jenis peristiwa ke target yang ditentukan untuk diproses, mungkin memperkenalkan kegagalan pemrosesan dan biaya pemrosesan tambahan.

Untuk membuat cakupan pola `Payment accepted` acara Anda hanya ke peristiwa, Anda ingin menentukan keduanya `source` dan`detail-type`, minimal:

```
{
  "detail-type": "Payment accepted",
  "source": "acme.payments"
  }
}
```

Anda juga dapat menentukan akun dan Wilayah dalam pola acara Anda, untuk membatasi lebih lanjut kapan peristiwa lintas akun atau lintas wilayah cocok dengan aturan ini.

```
{
  "account": "012345678910",
  "source": "acme.payments",
  "region": "AWS-Region",
  "detail-type": "Payment accepted"
}
```

Validasi pola acara

Untuk memastikan aturan sesuai dengan acara yang diinginkan, kami sangat menyarankan Anda memvalidasi pola acara Anda. Anda dapat memvalidasi pola acara menggunakan EventBridge konsol atau API:

- Di EventBridge konsol, Anda dapat membuat dan menguji pola acara sebagai bagian dari pembuatan aturan, atau secara terpisah dengan menggunakan Kotak Pasir.
- Anda dapat menguji pola acara Anda secara terprogram menggunakan tindakan. TestEventPattern

EventBridge Aturan Amazon

Anda menentukan apa yang EventBridge dilakukan dengan acara yang dikirimkan ke setiap bus acara. Untuk melakukan ini, Anda membuat aturan. Aturan menentukan peristiwa mana yang akan dikirim ke target mana yang akan diproses. Aturan tunggal dapat mengirim acara ke beberapa target, yang kemudian dijalankan secara paralel.

Anda dapat membuat dua jenis aturan:

- Aturan yang cocok dengan data acara

  Anda dapat membuat aturan yang cocok dengan peristiwa yang masuk berdasarkan kriteria data peristiwa (disebut pola peristiwa). Pola peristiwamendefinisikan struktur acara dan bidang yang cocok dengan aturan. Jika suatu peristiwa cocok dengan kriteria yang ditentukan dalam pola peristiwa, EventBridge kirimkan ke target yang Anda tentukan.

  Untuk informasi selengkapnya, lihat ???.
- Aturan yang berjalan sesuai jadwal

  Anda juga dapat membuat aturan yang mengirimkan peristiwa ke target yang ditentukan pada interval tertentu. Misalnya, untuk menjalankan Lambda fungsi secara berkala, Anda dapat membuat aturan untuk dijalankan sesuai jadwal.

> **Note**
>
> EventBridge menawarkan Amazon EventBridge Scheduler, penjadwal tanpa server yang memungkinkan Anda membuat, menjalankan, dan mengelola tugas dari satu layanan terpusat dan terkelola. EventBridge Scheduler sangat dapat disesuaikan, dan menawarkan skalabilitas yang ditingkatkan dibandingkan aturan EventBridge terjadwal, dengan serangkaian operasi dan layanan API target yang lebih luas. AWS Kami menyarankan Anda menggunakan EventBridge Scheduler untuk memanggil target pada jadwal. Untuk informasi selengkapnya, lihat ???.

Video berikut membahas dasar-dasar aturan: Apa itu aturan

Aturan yang EventBridge dikelola Amazon

Selain aturan yang Anda buat, AWS layanan dapat membuat dan mengelola EventBridge aturan di AWS akun Anda yang diperlukan untuk fungsi tertentu dalam layanan tersebut. Ini semuanya disebut aturan terkelola.

Saat layanan membuat aturan terkelola, layanan juga dapat membuat IAM kebijakan yang memberikan izin ke layanan tersebut untuk membuat aturan. Kebijakan IAM yang dibuat dengan cara ini memiliki jangkauan sempit dengan perizinan tingkat sumber daya untuk mengizinkan penciptaan aturan yang diperlukan saja.

Anda dapat menghapus aturan terkelola dengan menggunakan opsi Hapus paksa, tetapi Anda hanya harus menghapusnya jika Anda yakin bahwa layanan lain tidak lagi memerlukan aturan tersebut. Jika tidak, menghapus aturan terkelola menyebabkan fitur yang bergantung padanya berhenti bekerja.

Membuat EventBridge aturan Amazon yang bereaksi terhadap peristiwa

Untuk mengambil tindakan atas acara yang diterima oleh AmazonEventBridge, Anda dapat membuat aturan. Saat peristiwa cocok dengan pola peristiwa yang didefinisikan dalam aturan Anda, EventBridge mengirimkan peristiwa untuk target yang ditentukan dan memicu tindakan yang didefinisikan dalam aturan.

Video berikut mengeksplorasi pembuatan berbagai jenis aturan dan cara mengujinya: Belajar tentang aturan.

Gunakan prosedur berikut untuk membuat EventBridge aturan Amazon yang merespons peristiwa.

Buat aturan yang bereaksi terhadap peristiwa

Langkah-langkah berikut memandu Anda melalui cara membuat aturan yang EventBridge digunakan untuk mencocokkan peristiwa saat dikirim ke bus peristiwa yang ditentukan.

Langkah-langkah

- Tentukan aturan
- Membangun pola acara
- Pilih target
- Mengonfigurasi tag dan aturan ulasan

Tentukan aturan

Pertama, masukkan nama dan deskripsi untuk aturan Anda untuk mengidentifikasinya. Anda juga harus menentukan bus acara di mana aturan Anda mencari peristiwa yang cocok dengan pola acara.

Untuk menentukan detail aturan

1. Buka konsol Amazon EventBridge di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan Nama dan, opsional, Deskripsi untuk aturan.

Aturan tidak boleh memiliki nama yang sama dengan aturan lain di Wilayah AWS yang sama dan di bus peristiwa yang sama.

5. Untuk Bus peristiwa, pilih bus peristiwa yang akan dihubungkan dengan aturan ini. Jika Anda ingin aturan ini cocok dengan kejadian yang berasal dari akun Anda, pilih bus kejadian AWS default. Saat akun Layanan AWS Anda menghasilkan kejadian, akun Anda akan selalu masuk ke bus kejadian default akun Anda.
6. Untuk jenis Aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.

Membangun pola acara

Selanjutnya, bangun pola acara. Untuk melakukan ini, tentukan sumber acara, pilih dasar untuk pola acara, dan tentukan atribut dan nilai yang akan dicocokkan. Anda juga dapat membuat pola acara di JSON dan mengujinya terhadap contoh peristiwa.

Untuk membangun pola acara

1. Untuk Sumber acara, pilih AWSacara atau acara EventBridge mitra.
2. (Opsional) Di bagian Contoh peristiwa, pilih Contoh jenis peristiwa yang Anda inginkan untuk menguji pola acara Anda.

   Jenis kejadian contoh berikut ini tersedia:

   - AWSperistiwa - Pilih dari peristiwa yang dipancarkan dari didukung. Layanan AWS
   - EventBridgeacara mitra — Pilih dari acara yang dipancarkan dari layanan pihak ketiga yang mendukungEventBridge, seperti Salesforce.
   - Masukkan saya sendiri - Masukkan acara Anda sendiri dalam teks JSON.

     Anda juga dapat menggunakan acara AWS atau mitra sebagai titik awal untuk membuat acara kustom Anda sendiri.

     1. Pilih AWSacara atau acara EventBridge mitra.
     2. Gunakan dropdown Contoh peristiwa untuk memilih acara yang ingin Anda gunakan sebagai titik awal untuk acara kustom Anda.

        EventBridgemenampilkan contoh acara.
     3. Pilih Salin.

4. Pilih Enter my own for Event type.
5. Hapus contoh struktur peristiwa di panel pengeditan JSON, dan tempelkan peristiwa AWS atau mitra di tempatnya.
6. Edit acara JSON untuk membuat contoh acara Anda sendiri.

3. Pilih metode Creation. Anda dapat membuat pola acara dari EventBridge skema atau template, atau Anda dapat membuat pola acara kustom.

   Existing schema

   Untuk menggunakan EventBridge skema yang sudah ada untuk membuat pola peristiwa, lakukan hal berikut:

   1. Di bagian Creation method, untuk Method, pilih Use schema.
   2. Di bagian Pola acara, untuk jenis skema, pilih Pilih skema dari registri skema.
   3. Untuk registri Skema, pilih kotak dropdown dan masukkan nama registri skema, seperti. `aws.events` Anda juga dapat memilih opsi dari daftar dropdown yang muncul.
   4. Untuk Skema, pilih kotak dropdown dan masukkan nama skema yang akan digunakan. Sebagai contoh, `aws.s3@0bjectDeleted`. Anda juga dapat memilih opsi dari daftar dropdown yang muncul.
   5. Di bagian Model, pilih tombol Edit di samping atribut apa pun untuk membuka propertinya. Atur bidang Relationship dan Value sesuai kebutuhan, lalu pilih Set untuk menyimpan atribut.

      > **Note**
      >
      > Untuk informasi tentang definisi atribut, pilih ikon Info di samping nama atribut. Untuk referensi tentang cara mengatur properti atribut dalam acara Anda, buka bagian Catatan pada kotak dialog atribut properti.
      > Untuk menghapus properti atribut, pilih tombol Edit untuk atribut itu, lalu pilih Hapus.

   6. Pilih Buat pola acara di JSON untuk menghasilkan dan memvalidasi pola acara Anda sebagai teks JSON.
   7. (Opsional) Untuk menguji kejadian sampel terhadap pola pengujian Anda, pilih Pola uji.

EventBridgemenampilkan kotak pesan yang menyatakan apakah acara sampel Anda cocok dengan pola acara.

Anda juga dapat memilih salah satu opsi berikut ini:

- Salin - Salin pola peristiwa ke clipboard perangkat Anda.
- Prettify - Membuat teks JSON lebih mudah dibaca dengan menambahkan jeda baris, tab, dan spasi.

Custom schema

Untuk menulis skema kustom dan mengubahnya menjadi pola acara, lakukan hal berikut:

1. Di bagian Creation method, untuk Method, pilih Use schema.
2. Di bagian Event pattern, untuk tipe Skema, pilih Enter schema.
3. Masukkan skema Anda ke kotak teks. Anda harus memformat skema sebagai teks JSON yang valid.
4. Di bagian Model, pilih tombol Edit di samping atribut apa pun untuk membuka propertinya. Atur bidang Relationship dan Value sesuai kebutuhan, lalu pilih Set untuk menyimpan atribut.

   > **Note**
   >
   > Untuk informasi tentang definisi atribut, pilih ikon Info di samping nama atribut.
   > Untuk referensi tentang cara mengatur properti atribut dalam acara Anda, buka bagian Catatan pada kotak dialog atribut properti.
   > Untuk menghapus properti atribut, pilih tombol Edit untuk atribut itu, lalu pilih Hapus.

5. Pilih Buat pola acara di JSON untuk menghasilkan dan memvalidasi pola acara Anda sebagai teks JSON.
6. (Opsional) Untuk menguji kejadian sampel terhadap pola pengujian Anda, pilih Pola uji.

   EventBridgemenampilkan kotak pesan yang menyatakan apakah acara sampel Anda cocok dengan pola acara.

   Anda juga dapat memilih salah satu opsi berikut ini:

- Salin - Salin pola peristiwa ke clipboard perangkat Anda.
- Prettify - Membuat teks JSON lebih mudah dibaca dengan menambahkan jeda baris, tab, dan spasi.

Event pattern

Untuk menulis pola peristiwa kustom dalam format JSON, lakukan hal berikut:

1. Di bagian Creation method, untuk Method, pilih Custom pattern (JSON editor).
2. Untuk pola acara, masukkan pola acara kustom Anda dalam teks berformat JSON.
3. (Opsional) Untuk menguji kejadian sampel terhadap pola pengujian Anda, pilih Pola uji.

   EventBridgemenampilkan kotak pesan yang menyatakan apakah acara sampel Anda cocok dengan pola acara.

   Anda juga dapat memilih salah satu opsi berikut ini:

   - Salin - Salin pola peristiwa ke clipboard perangkat Anda.
   - Prettify - Membuat teks JSON lebih mudah dibaca dengan menambahkan jeda baris, tab, dan spasi.
   - Bentuk pola acara - Membuka pola acara di Pattern Builder. Jika pola tidak dapat diberikan dalam Pattern Builder apa adanya, EventBridge memperingatkan Anda sebelum membuka Pattern Builder.
4. Pilih Selanjutnya.

Pilih target

Pilih satu atau lebih target untuk menerima peristiwa yang sesuai dengan pola yang ditentukan. Target dapat mencakup bus EventBridge acara, tujuan EventBridge API, termasuk mitra SaaS seperti Salesforce, atau lainnya. Layanan AWS

Untuk memilih target

1. Untuk Jenis target, pilih salah satu jenis target berikut:

   Event bus

   Untuk memilih bus EventBridge acara, pilih bus EventBridge acara, lalu lakukan hal berikut:

- Untuk menggunakan bus peristiwa yang Wilayah AWS sama dengan aturan ini:
  1. Pilih Bus acara di akun dan Wilayah yang sama.
  2. Untuk bus Acara untuk target, pilih kotak dropdown dan masukkan nama bus acara. Anda juga dapat memilih bus peristiwa dari daftar tarik-turun.

     Untuk informasi selengkapnya, lihat ???.
- Untuk menggunakan bus peristiwa di akun lain Wilayah AWS atau akun seperti aturan ini:
  1. Pilih Bus peristiwa di akun atau Wilayah yang berbeda.
  2. Untuk bus Event sebagai target, masukkan ARN bus acara yang ingin Anda gunakan.

     Untuk informasi selengkapnya, lihat:
     - ???
     - ???

API destination

Untuk menggunakan tujuan EventBridge API, pilih tujuan EventBridge API, lalu lakukan salah satu hal berikut:

- Untuk menggunakan tujuan API yang ada, pilih Gunakan tujuan API yang ada. Kemudian pilih tujuan API dari daftar tarik-turun.
- Untuk membuat tujuan API baru, pilih Buat tujuan API baru. Kemudian, berikan rincian berikut untuk tujuan:
  - Nama - Masukkan nama untuk tujuan.

    Nama harus unik di dalam AndaAkun AWS. Nama dapat memiliki hingga 64 karakter. Karakter yang valid adalah A-Z, a-z, 0-9, dan. _ - (tanda hubung).
  - (Opsional) Deskripsi - Masukkan deskripsi untuk tujuan.

    Deskripsi dapat memiliki hingga 512 karakter.
  - Titik akhir tujuan API - Titik akhir URL untuk target.

    URL endpoint harus dimulai dengan**`https`**. Anda dapat menyertakan **`*`** sebagai parameter path wildcard. Anda dapat mengatur parameter jalur dari `HttpParameters` atribut target.

- Metode HTTP - Pilih metode HTTP yang digunakan saat Anda memanggil titik akhir.
- (Opsional) Batas tingkat pemanggilan per detik — Masukkan jumlah maksimum pemanggilan yang diterima untuk setiap detik untuk tujuan ini.

  Nilai ini harus lebih besar dari nol. Secara default, nilai ini diatur ke 300.
- Koneksi - Pilih untuk menggunakan koneksi baru atau yang sudah ada:
  - Untuk menggunakan koneksi yang ada, pilih Gunakan koneksi yang ada dan pilih koneksi dari daftar dropdown.
  - Untuk membuat sambungan baru untuk tujuan ini pilih Buat sambungan baru, lalu tentukan nama sambungan, jenis tujuan, dan jenis otorisasi. Anda juga dapat menambahkan nilai Deskripsi opsional untuk koneksi ini.

Untuk informasi selengkapnya, lihat ???.

Layanan AWS

Untuk menggunakanLayanan AWS, pilih Layanan AWS, lalu lakukan hal berikut:

1. Untuk Pilih target, pilih Layanan AWS untuk digunakan sebagai target. Berikan informasi yang diminta untuk layanan yang Anda pilih.

   > **Note**
   >
   > Bidang yang ditampilkan bervariasi tergantung pada layanan yang dipilih. Untuk informasi selengkapnya tentang target yang tersedia, lihatTarget tersedia di EventBridge konsol.

2. Untuk sebagian besar tipe target, EventBridge membutuhkan izin untuk mengirim kejadian ke target. Dalam kasus ini, EventBridge dapat membuat IAM role yang diperlukan bagi aturan Anda untuk berjalan.

   Untuk Execution role, lakukan salah satu hal berikut:

   - Untuk membuat peran eksekusi baru untuk aturan ini:

     a. Pilih Buat peran baru untuk sumber daya khusus ini.

     b. Entah masukkan nama untuk peran eksekusi ini, atau gunakan nama yang dihasilkan olehEventBridge.

- Untuk menggunakan peran eksekusi yang ada untuk aturan ini:

  a. Pilih Gunakan peran yang ada.

  b. Masukkan atau pilih nama peran eksekusi yang akan digunakan dari daftar dropdown.

3. (Opsional) Untuk Pengaturan tambahan, tentukan salah satu pengaturan opsional yang tersedia untuk jenis target Anda:

   Event bus

   (Opsional) Untuk Antrean surat mati, pilih apakah akan Anda menggunakan antrean Amazon SQS standar sebagai antrean surat mati. EventBridgemengirimkan peristiwa yang mencocokkan aturan ini dengan antrean surat mati jika tidak berhasil dikirim ke target. Lakukan salah satu dari berikut:

   - Pilih Tidak ada untuk tidak menggunakan antrean surat mati.
   - Pilih Pilihan antrean Amazon SQS di akun AWS saat ini untuk digunakan sebagai antrean surat mati kemudian pilih antrean yang akan digunakan dari daftar menurun.
   - Pilih Pilihan antrean Amazon SQS di akun AWS lainnya sebagai antrean surat mati dan kemudian masukkan ARN antrean untuk menggunakannya. Anda harus melampirkan kebijakan berbasis sumber daya ke antrean yang memberikan izin EventBridge untuk mengirim pesan padanya.

     Untuk informasi selengkapnya, lihat Memberikan izin untuk antrean surat mati.

   API destination

   1. (Opsional) Untuk Konfigurasi input target, pilih bagaimana Anda ingin menyesuaikan teks yang dikirim ke target untuk acara yang cocok. Pilih salah satu dari berikut:
      - Peristiwa yang cocok - EventBridge mengirimkan seluruh acara sumber asli ke target. Ini adalah default.
      - Bagian dari peristiwa yang cocok - EventBridge hanya mengirimkan bagian tertentu dari acara sumber asli ke target.

        Di bawah Tentukan bagian dari peristiwa yang cocok, tentukan jalur JSON yang mendefinisikan bagian dari acara yang ingin Anda kirim EventBridge ke target.
      - Konstan (teks JSON) - hanya EventBridge mengirimkan teks JSON yang ditentukan ke target. Tidak ada bagian dari acara sumber asli yang dikirim.

Di bawah Tentukan konstanta di JSON, tentukan teks JSON yang ingin Anda kirim EventBridge ke target, bukan peristiwa.

- Transformator input - Konfigurasikan transformator input untuk menyesuaikan teks yang ingin Anda EventBridge kirim ke target. Untuk informasi selengkapnya, lihat ???.

  a. Pilih Konfigurasikan transformator input.

  b. Konfigurasikan transformator input mengikuti langkah-langkah di???.

2. (Opsional) Di bawah kebijakan Coba lagi, tentukan bagaimana EventBridge seharusnya mencoba lagi mengirim peristiwa ke target setelah terjadi kesalahan.
   - Usia maksimum acara - Masukkan jumlah waktu maksimum (dalam jam, menit, dan detik) EventBridge untuk mempertahankan kejadian yang belum diproses. Pengaturan default-nya adalah 24 jam.
   - Coba lagi upaya - Masukkan jumlah maksimum kali EventBridge harus mencoba lagi mengirim peristiwa ke target setelah terjadi kesalahan. Defaultnya adalah 185 kali.
3. (Opsional) Untuk Antrean surat mati, pilih apakah akan Anda menggunakan antrean Amazon SQS standar sebagai antrean surat mati. EventBridgemengirimkan peristiwa yang mencocokkan aturan ini dengan antrean surat mati jika tidak berhasil dikirim ke target. Lakukan salah satu dari berikut:
   - Pilih Tidak ada untuk tidak menggunakan antrean surat mati.
   - Pilih Pilihan antrean Amazon SQS di akun AWS saat ini untuk digunakan sebagai antrean surat mati kemudian pilih antrean yang akan digunakan dari daftar menurun.
   - Pilih Pilihan antrean Amazon SQS di akun AWS lainnya sebagai antrean surat mati dan kemudian masukkan ARN antrean untuk menggunakannya. Anda harus melampirkan kebijakan berbasis sumber daya ke antrean yang memberikan izin EventBridge untuk mengirim pesan padanya.

     Untuk informasi selengkapnya, lihat Memberikan izin untuk antrean surat mati.

AWS service

Perhatikan bahwa EventBridge mungkin tidak menampilkan semua bidang berikut untuk AWS layanan tertentu.

1. (Opsional) Untuk Konfigurasi input target, pilih bagaimana Anda ingin menyesuaikan teks yang dikirim ke target untuk acara yang cocok. Pilih salah satu dari berikut:

- Peristiwa yang cocok - EventBridge mengirimkan seluruh acara sumber asli ke target. Ini adalah default.
- Bagian dari peristiwa yang cocok - EventBridge hanya mengirimkan bagian tertentu dari acara sumber asli ke target.

  Di bawah Tentukan bagian dari peristiwa yang cocok, tentukan jalur JSON yang mendefinisikan bagian dari acara yang ingin Anda kirim EventBridge ke target.
- Konstan (teks JSON) - hanya EventBridge mengirimkan teks JSON yang ditentukan ke target. Tidak ada bagian dari acara sumber asli yang dikirim.

  Di bawah Tentukan konstanta di JSON, tentukan teks JSON yang ingin Anda kirim EventBridge ke target, bukan peristiwa.
- Transformator input - Konfigurasikan transformator input untuk menyesuaikan teks yang ingin Anda EventBridge kirim ke target. Untuk informasi selengkapnya, lihat ???.

  a. Pilih Konfigurasikan transformator input.

  b. Konfigurasikan transformator input mengikuti langkah-langkah di???.

2. (Opsional) Di bawah kebijakan Coba lagi, tentukan bagaimana EventBridge seharusnya mencoba lagi mengirim peristiwa ke target setelah terjadi kesalahan.
   - Usia maksimum acara - Masukkan jumlah waktu maksimum (dalam jam, menit, dan detik) EventBridge untuk mempertahankan kejadian yang belum diproses. Pengaturan default-nya adalah 24 jam.
   - Coba lagi upaya - Masukkan jumlah maksimum kali EventBridge harus mencoba lagi mengirim peristiwa ke target setelah terjadi kesalahan. Defaultnya adalah 185 kali.
3. (Opsional) Untuk Antrean surat mati, pilih apakah akan Anda menggunakan antrean Amazon SQS standar sebagai antrean surat mati. EventBridgemengirimkan peristiwa yang mencocokkan aturan ini dengan antrean surat mati jika tidak berhasil dikirim ke target. Lakukan salah satu dari berikut:
   - Pilih Tidak ada untuk tidak menggunakan antrean surat mati.
   - Pilih Pilihan antrean Amazon SQS di akun AWS saat ini untuk digunakan sebagai antrean surat mati kemudian pilih antrean yang akan digunakan dari daftar menurun.
   - Pilih Pilihan antrean Amazon SQS di akun AWS lainnya sebagai antrean surat mati dan kemudian masukkan ARN antrean untuk menggunakannya. Anda harus melampirkan kebijakan berbasis sumber daya ke antrean yang memberikan izin EventBridge untuk mengirim pesan padanya.

Untuk informasi selengkapnya, lihat Memberikan izin untuk antrean surat mati.

4. (Opsional) Pilih Tambahkan target lainnya untuk menambahkan target lain untuk aturan ini.
5. Pilih Selanjutnya.

Perhatikan bahwa EventBridge mungkin tidak menampilkan semua bidang berikut untuk AWS layanan tertentu.

Mengonfigurasi tag dan aturan ulasan

Terakhir, masukkan tag yang diinginkan untuk aturan, lalu tinjau dan buat aturan.

Untuk mengkonfigurasi tag, dan meninjau dan membuat aturan

1. (Opsional) Masukkan satu atau lebih tanda untuk aturan. Untuk informasi selengkapnya, lihat EventBridge Tag Amazon.
2. Pilih Next (Berikutnya).
3. Tinjau detail untuk aturan baru. Untuk membuat perubahan pada bagian mana pun, pilih tombol Edit di sebelah bagian itu.

   Jika puas dengan detail aturan, pilih Buat aturan.

Menggunakan Amazon EventBridge Scheduler dengan Amazon EventBridge

Amazon EventBridge Scheduler adalah penjadwal tanpa server yang memungkinkan Anda membuat, menjalankan, dan mengelola tugas dari satu layanan terpusat dan terkelola. Dengan EventBridge Scheduler, Anda dapat membuat jadwal menggunakan ekspresi cron dan rate untuk pola berulang, atau mengonfigurasi pemanggilan satu kali. Anda dapat mengatur jendela waktu fleksibel untuk pengiriman, menentukan batas coba lagi, dan mengatur waktu retensi maksimum untuk pemanggilan API yang gagal.

EventBridge Scheduler sangat dapat disesuaikan, dan menawarkan skalabilitas yang ditingkatkan dibandingkan aturan EventBridge terjadwal, dengan serangkaian operasi dan layanan API target yang lebih luas. AWS Kami menyarankan Anda menggunakan EventBridge Scheduler untuk memanggil target pada jadwal.

Topik

- Mengatur peran eksekusi
- Buat jadwal
- Sumber daya terkait

Mengatur peran eksekusi

Saat Anda membuat jadwal baru, EventBridge Scheduler harus memiliki izin untuk menjalankan operasi API targetnya atas nama Anda. Anda memberikan izin ini ke EventBridge Scheduler menggunakan peran eksekusi. Kebijakan izin yang Anda lampirkan ke peran eksekusi jadwal menentukan izin yang diperlukan. Izin ini bergantung pada API target yang ingin Anda panggil EventBridge Scheduler.

Bila Anda menggunakan konsol EventBridge Scheduler untuk membuat jadwal, seperti dalam prosedur berikut, EventBridge Scheduler secara otomatis mengatur peran eksekusi berdasarkan target yang Anda pilih. Jika Anda ingin membuat jadwal menggunakan salah satu SDK EventBridge Penjadwal, atau AWS CLIAWS CloudFormation, Anda harus memiliki peran eksekusi yang ada yang memberikan izin yang diperlukan EventBridge Penjadwal untuk memanggil target. Untuk informasi selengkapnya tentang mengatur peran eksekusi secara manual untuk jadwal Anda, lihat Menyiapkan peran eksekusi di Panduan Pengguna EventBridge Penjadwal.

Buat jadwal

Untuk membuat jadwal dengan menggunakan konsol

1. Buka konsol Amazon EventBridge Scheduler di https://console.aws.amazon.com/scheduler/home.
2. Pada halaman Jadwal, pilih Buat jadwal.
3. Pada halaman Tentukan detail jadwal, di bagian Nama jadwal dan deskripsi, lakukan hal berikut:

   a. Untuk nama Jadwal, masukkan nama untuk jadwal Anda. Sebagai contoh, **`MyTestSchedule`**.

   b. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk jadwal Anda. Sebagai contoh, **`My first schedule`**.

   c. Untuk grup Jadwal, pilih grup jadwal dari daftar dropdown. Jika Anda tidak memiliki grup, pilih default. Untuk membuat grup jadwal, pilih buat jadwal Anda sendiri.

   Anda menggunakan grup jadwal untuk menambahkan tag ke grup jadwal.

4. • Pilih opsi jadwal Anda.

Kejadian	Lakukan ini...	
Jadwal satu kali Jadwal satu kali memanggil target hanya sekali pada tanggal dan waktu yang Anda tentukan.	Untuk tanggal dan waktu, lakukan hal berikut: • Masukkan tanggal yang valid dalam `YYYY/MM/DD` format. • Masukkan stempel waktu dalam format 24 jam`hh:mm`. • Untuk Timezone, pilih zona waktu.	
Jadwal berulang Jadwal berulang memanggil target pada	a. Untuk jenis Jadwal, lakukan salah satu hal berikut:	

Kejadian	Lakukan ini...	
tingkat yang Anda tentukan menggunakan cron ekspresi atau ekspresi tingkat.	• Untuk menggunakan ekspresi cron untuk menentukan jadwal, pilih Jadwal berbasis Cron dan masukkan ekspresi cron. • Untuk menggunakan ekspresi laju untuk menentukan jadwal, pilih Jadwal berbasis tarif dan masukkan ekspresi laju. Untuk informasi selengkapnya tentang ekspresi cron dan rate, lihat Menjadwalkan jenis pada EventBridge Scheduler di Panduan Pengguna EventBridge Penjadwal Amazon. b. Untuk jendela waktu Fleksibel, pilih Nonaktif untuk mematikan opsi, atau pilih salah satu jendela waktu yang telah ditentukan sebelumnya. Misalnya, jika Anda memilih 15 menit dan Anda menetapkan jadwal berulang untuk memanggil targetnya	

Kejadian	Lakukan ini...	
	setiap jam sekali, jadwal berjalan dalam 15 menit setelah dimulainya setiap jam.	

5. (Opsional) Jika Anda memilih Jadwal berulang pada langkah sebelumnya, di bagian Jangka Waktu, lakukan hal berikut:

   a. Untuk Timezone, pilih zona waktu.

   b. Untuk Tanggal dan waktu mulai, masukkan tanggal yang valid dalam `YYYY/MM/DD` format, lalu tentukan stempel waktu dalam format 24 jam`hh:mm`.

   c. Untuk Tanggal dan waktu berakhir, masukkan tanggal yang valid dalam `YYYY/MM/DD` format, lalu tentukan stempel waktu dalam format 24 jam`hh:mm`.

6. Pilih Selanjutnya.

7. Pada halaman Select target, pilih operasi AWS API yang dipanggil EventBridge Scheduler:

   a. Untuk API Target, pilih Target Templated.

   b. Pilih Amazon EventBridge PutEvents.

   c. Di bawah PutEvents, tentukan yang berikut ini:

      - Untuk bus EventBridge acara, pilih bus acara dari menu drop-down. Sebagai contoh, **`default`**.

        Anda juga dapat membuat bus acara baru di EventBridge konsol dengan memilih Buat bus acara baru.

      - Untuk tipe Detail, masukkan jenis detail acara yang ingin Anda cocokkan. Sebagai contoh, **`Object Created`**.

      - Untuk Sumber, masukkan nama layanan yang merupakan sumber acara.

        Untuk acara AWS layanan, tentukan awalan layanan sebagai sumber. Jangan sertakan `aws.` awalan. Misalnya, untuk acara Amazon S3, masukkan. **`s3`**

        Untuk menentukan awalan layanan, lihat Tabel kunci kondisi di Referensi Otorisasi Layanan. Untuk informasi selengkapnya tentang sumber dan nilai peristiwa tipe detail, lihat. ???

- (Opsional): Untuk Detail, masukkan pola acara untuk memfilter lebih lanjut peristiwa yang dikirim oleh EventBridge Scheduler. EventBridge

  Untuk informasi selengkapnya, lihat ???.

8. Pilih Next (Berikutnya).
9. Pada halaman Pengaturan, lakukan hal berikut:

   a. Untuk mengaktifkan jadwal, di bawah Status jadwal, alihkan Aktifkan jadwal.

   b. Untuk mengonfigurasi kebijakan coba lagi untuk jadwal Anda, di bawah Kebijakan Coba ulang dan antrian surat mati (DLQ), lakukan hal berikut:

      - Beralih Coba Lagi.
      - Untuk usia maksimum acara, masukkan jam maksimum dan min yang harus disimpan oleh EventBridge Scheduler untuk menyimpan acara yang belum diproses.
      - Waktu maksimum adalah 24 jam.
      - Untuk percobaan ulang Maksimum, masukkan jumlah maksimum kali EventBridge Scheduler mencoba ulang jadwal jika target mengembalikan kesalahan.

        Nilai maksimum adalah 185 percobaan ulang.

      Dengan kebijakan coba lagi, jika jadwal gagal memanggil targetnya, EventBridge Scheduler menjalankan kembali jadwal. Jika dikonfigurasi, Anda harus mengatur waktu retensi maksimum dan mencoba ulang untuk jadwal.

   c. Pilih tempat EventBridge Scheduler menyimpan acara yang tidak terkirim.

Opsi antrian surat mati (DLQ)	Lakukan ini...	
Jangan simpan	Pilih Tidak Ada.	
Simpan acara di tempat yang sama Akun AWS di mana Anda membuat jadwal	a. Pilih Pilih antrian Amazon SQS di saya Akun AWS sebagai DLQ. b. Pilih Nama Sumber Daya Amazon (ARN)	

Opsi antrian surat mati (DLQ)	Lakukan ini...
	dari antrian Amazon SQS.
Simpan acara di tempat yang berbeda Akun AWS dari tempat Anda membuat jadwal	a. Pilih Tentukan antrean Amazon SQS di lain Akun AWS sebagai DLQ. b. Masukkan Nama Sumber Daya Amazon (ARN) dari antrian Amazon SQS.

d. Untuk menggunakan kunci yang dikelola pelanggan untuk mengenkripsi input target Anda, di bawah Enkripsi, pilih Sesuaikan pengaturan enkripsi (lanjutan).

   Jika Anda memilih opsi ini, masukkan ARN kunci KMS yang ada atau pilih AWS KMS key Buat untuk menavigasi ke AWS KMS konsol. Untuk informasi selengkapnya tentang cara EventBridge Scheduler mengenkripsi data Anda saat istirahat, lihat Enkripsi saat istirahat di Panduan Pengguna EventBridge Penjadwal Amazon.

e. Agar EventBridge Scheduler membuat peran eksekusi baru untuk Anda, pilih Buat peran baru untuk jadwal ini. Kemudian, masukkan nama untuk nama Peran. Jika Anda memilih opsi ini, EventBridge Scheduler melampirkan izin yang diperlukan untuk target template Anda ke peran.

10. Pilih Selanjutnya.
11. Di halaman Tinjau dan buat jadwal, tinjau detail jadwal Anda. Di setiap bagian, pilih Edit untuk kembali ke langkah itu dan mengedit detailnya.
12. Pilih Buat jadwal.

    Anda dapat melihat daftar jadwal baru dan yang sudah ada di halaman Jadwal. Di bawah kolom Status, verifikasi bahwa jadwal baru Anda Diaktifkan.

Sumber daya terkait

Untuk informasi selengkapnya tentang EventBridge Scheduler, lihat berikut ini:

- EventBridge Panduan Pengguna Penjadwal
- EventBridge Referensi API Scheduler
- EventBridge Harga Penjadwal

Membuat EventBridge aturan Amazon yang berjalan sesuai jadwal

Aturan dapat berjalan sebagai respons terhadap suatu peristiwa, atau pada interval waktu tertentu. Sebagai contoh, untuk menjalankan AWS Lambdafungsi secara berkala, Anda dapat membuat aturan untuk berjalan sesuai jadwal.

> **Note**
>
> EventBridge menawarkan Amazon EventBridge Scheduler, penjadwal tanpa server yang memungkinkan Anda membuat, menjalankan, dan mengelola tugas dari satu layanan terpusat dan terkelola. EventBridge Scheduler sangat dapat disesuaikan, dan menawarkan skalabilitas yang ditingkatkan dibandingkan aturan EventBridge terjadwal, dengan serangkaian operasi dan layanan API target yang lebih luas. AWS
> Kami menyarankan Anda menggunakan EventBridge Scheduler untuk memanggil target pada jadwal. Untuk informasi selengkapnya, lihat ???.

Di EventBridge, Anda dapat membuat dua jenis aturan terjadwal:

- Aturan yang berjalan pada tingkat reguler

  EventBridge menjalankan aturan ini secara berkala; misalnya, setiap 20 menit.

  Untuk menentukan tingkat untuk aturan terjadwal, Anda menentukan ekspresi tingkat.
- Aturan yang berjalan pada waktu tertentu

  EventBridge menjalankan aturan ini pada waktu dan tanggal tertentu; misalnya, 8:00 a.m. PST pada hari Senin pertama setiap bulan.

  Untuk menentukan waktu dan tanggal aturan terjadwal berjalan, Anda menentukan ekspresi cron.

Ekspresi tingkat lebih sederhana untuk didefinisikan, sementara ekspresi cron menawarkan kontrol jadwal terperinci. Sebagai contoh, dengan ekspresi cron, Anda dapat menentukan aturan yang

berjalan pada waktu tertentu pada hari tertentu setiap minggu atau bulan. Sebaliknya, ekspresi tingkat menjalankan aturan pada tingkat reguler, seperti sekali setiap jam atau sekali setiap hari.

Semua acara yang dijadwalkan menggunakan zona waktu UTC+0, dan presisi minimum untuk jadwal adalah satu menit.

> **Note**
>
> EventBridge tidak memberikan presisi tingkat kedua dalam ekspresi jadwal. Resolusi terbaik yang menggunakan ekspresi cron adalah satu menit. Karena sifat terdistribusi EventBridge dan layanan target, mungkin ada penundaan beberapa detik antara waktu aturan terjadwal dipicu dan waktu layanan target menjalankan sumber daya target.

Video berikut memberikan ikhtisar tugas penjadwalan: Membuat tugas terjadwal dengan EventBridge

Topik

- Buat aturan yang berjalan sesuai jadwal
- Referensi ekspresi cron
- Referensi ekspresi nilai

Buat aturan yang berjalan sesuai jadwal

Langkah-langkah berikut memandu Anda melalui cara membuat EventBridge aturan yang berjalan pada jadwal reguler.

> **Note**
>
> Anda hanya dapat membuat aturan terjadwal dengan menggunakan bus peristiwa default.

Langkah-langkah

- Tentukan aturannya
- Tentukan jadwal
- Pilih target

- Konfigurasikan tag dan aturan ulasan

Tentukan aturannya

Pertama, masukkan nama dan deskripsi untuk aturan Anda untuk mengidentifikasinya.

Untuk menentukan detail aturan

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan Nama dan, secara opsional, Deskripsi untuk aturan.

   Aturan tidak boleh memiliki nama yang sama dengan aturan lain di Wilayah AWS yang sama dan di bus peristiwa yang sama.
5. Untuk bus acara, pilih bus acara default. Anda hanya dapat membuat aturan terjadwal dengan menggunakan bus peristiwa default.
6. Agar aturan berlaku segera setelah Anda membuatnya, pastikan opsi Aktifkan aturan pada bus acara yang dipilih diaktifkan.
7. Untuk jenis Aturan, pilih Jadwal.

   Pada titik ini, Anda dapat memilih untuk melanjutkan dengan membuat aturan yang berjalan sesuai jadwal, atau menggunakan Amazon EventBridge Scheduler.
8. Pilih bagaimana Anda ingin melanjutkan:

   - Gunakan EventBridge Scheduler untuk membuat jadwal Anda

     > **Note**
     >
     > EventBridge Scheduler adalah penjadwal tanpa server yang memungkinkan Anda membuat, menjalankan, dan mengelola tugas dari satu layanan terpusat yang dikelola. Ini menyediakan fungsionalitas penjadwalan satu kali dan berulang yang independen dari bus dan aturan acara. EventBridge Scheduler sangat dapat disesuaikan, dan menawarkan skalabilitas yang ditingkatkan dibandingkan aturan EventBridge terjadwal, dengan serangkaian operasi dan layanan API target yang lebih luas. AWS

Kami menyarankan Anda menggunakan EventBridge Scheduler untuk memanggil target pada jadwal. Untuk informasi selengkapnya, lihat Apa itu Amazon EventBridge Scheduler? di Panduan Pengguna EventBridge Penjadwal Amazon.

1. Pilih Lanjutkan di EventBridge Scheduler

   EventBridge membuka konsol EventBridge Scheduler ke halaman Buat jadwal.

2. Buat jadwal di konsol EventBridge Scheduler.

- Lanjutkan menggunakan EventBridge untuk membuat aturan terjadwal untuk bus acara default
  1. Pilih Lanjutkan untuk membuat aturan.

Tentukan jadwal

Selanjutnya, tentukan pola jadwal.

Untuk menentukan pola jadwal

1. Untuk pola Jadwal, pilih apakah Anda ingin jadwal berjalan pada waktu tertentu, atau dengan tarif reguler:

   Specific time

   1. Pilih Jadwal berbutir halus yang berjalan pada waktu tertentu, seperti pukul 8:00 pagi. PST pada hari Senin pertama setiap bulan.
   2. Untuk ekspresi Cron, tentukan bidang untuk menentukan ekspresi cron yang EventBridge harus digunakan untuk menentukan kapan harus menjalankan aturan terjadwal ini.

      Setelah Anda menentukan semua bidang, EventBridge menampilkan sepuluh tanggal berikutnya kapan EventBridge akan mengeksekusi aturan terjadwal ini. Anda dapat memilih apakah akan menampilkan tanggal tersebut di UTC atau zona waktu lokal.

      Untuk informasi lebih lanjut tentang membangun ekspresi cron, lihat. ???

   Regular rate

   1. Pilih jadwal yang berjalan dengan tarif reguler, seperti setiap 10 menit.

2. Untuk ekspresi Rate, tentukan bidang Nilai dan Unit untuk menentukan tingkat di mana EventBridge harus menjalankan aturan terjadwal ini.

   Untuk informasi lebih lanjut tentang membangun ekspresi laju, lihat???.

2. Pilih Selanjutnya.

Pilih target

Pilih satu atau beberapa target untuk menerima peristiwa yang cocok dengan pola yang ditentukan. Target dapat mencakup bus EventBridge acara, tujuan EventBridge API, termasuk mitra SaaS seperti Salesforce, atau lainnya. Layanan AWS

Untuk memilih target

1. Untuk jenis Target, pilih salah satu jenis target berikut:

   Event bus

   Untuk memilih bus EventBridge acara, pilih bus EventBridge acara, lalu lakukan hal berikut:

   - Untuk menggunakan bus acara yang Wilayah AWS sama dengan aturan ini:
     1. Pilih Bus acara di akun dan Wilayah yang sama.
     2. Untuk bus acara untuk target, pilih kotak dropdown dan masukkan nama bus acara. Anda juga dapat memilih bus acara dari daftar dropdown.

        Untuk informasi selengkapnya, lihat ???.
   - Untuk menggunakan bus acara di akun yang berbeda Wilayah AWS atau sebagai aturan ini:
     1. Pilih bus Acara di akun atau Wilayah yang berbeda.
     2. Untuk bus Event sebagai target, masukkan ARN bus acara yang ingin Anda gunakan.

        Untuk informasi selengkapnya, lihat:

        - ???
        - ???

API destination

Untuk menggunakan tujuan EventBridge API, pilih tujuan EventBridge API, lalu lakukan salah satu hal berikut:

- Untuk menggunakan tujuan API yang ada, pilih Gunakan tujuan API yang ada. Kemudian pilih tujuan API dari daftar dropdown.
- Untuk membuat tujuan API baru, pilih Buat tujuan API baru. Kemudian, berikan rincian berikut untuk tujuan:
  - Nama — Masukkan nama untuk tujuan.

    Nama harus unik di dalam diri AndaAkun AWS. Nama dapat memiliki hingga 64 karakter. Karakter yang valid adalah A-Z, a-z, 0-9, dan. _ - (tanda hubung).
  - (Opsional) Deskripsi — Masukkan deskripsi untuk tujuan.

    Deskripsi dapat memiliki hingga 512 karakter.
  - Titik akhir tujuan API — Titik akhir URL untuk target.

    URL endpoint harus dimulai dengan**`https`**. Anda dapat menyertakan wildcard **`*`** sebagai parameter jalur. Anda dapat mengatur parameter jalur dari `HttpParameters` atribut target.
  - Metode HTTP - Pilih metode HTTP yang digunakan saat Anda memanggil titik akhir.
  - (Opsional) Batas tingkat pemanggilan per detik - Masukkan jumlah maksimum pemanggilan yang diterima untuk setiap detik untuk tujuan ini.

    Nilai ini harus lebih besar dari nol. Secara default, nilai ini diatur ke 300.
  - Koneksi — Pilih untuk menggunakan koneksi baru atau yang sudah ada:
    - Untuk menggunakan koneksi yang ada, pilih Gunakan koneksi yang ada dan pilih koneksi dari daftar tarik-turun.
    - Untuk membuat koneksi baru untuk tujuan ini pilih Buat koneksi baru, lalu tentukan Nama koneksi, tipe Tujuan, dan jenis Otorisasi. Anda juga dapat menambahkan Deskripsi opsional untuk koneksi ini.

Untuk informasi selengkapnya, lihat ???.

Layanan AWS

Untuk menggunakanLayanan AWS, pilih Layanan AWS, lalu lakukan hal berikut:

1. Untuk Pilih target, pilih Layanan AWS untuk digunakan sebagai target. Berikan informasi yang diminta untuk layanan yang Anda pilih.

> **Note**
>
> Bidang yang ditampilkan bervariasi tergantung pada layanan yang dipilih. Untuk informasi selengkapnya tentang target yang tersedia, lihatTarget tersedia di EventBridge konsol.

2. Untuk sebagian besar tipe target, EventBridge membutuhkan izin untuk mengirim kejadian ke target. Dalam kasus ini, EventBridge dapat membuat peran IAM yang diperlukan agar aturan Anda berjalan.

   Untuk peran Eksekusi, lakukan salah satu hal berikut:

   - Untuk membuat peran eksekusi baru untuk aturan ini:

     a. Pilih Buat peran baru untuk sumber daya khusus ini.

     b. Masukkan nama untuk peran eksekusi ini, atau gunakan nama yang dihasilkan oleh EventBridge.

   - Untuk menggunakan peran eksekusi yang ada untuk aturan ini:

     a. Pilih Gunakan peran yang ada.

     b. Masukkan atau pilih nama peran eksekusi yang akan digunakan dari daftar dropdown.

3. (Opsional) Untuk pengaturan Tambahan, tentukan salah satu pengaturan opsional yang tersedia untuk jenis target Anda:

   Event bus

   (Opsional) Untuk antrian Dead-letter, pilih apakah akan menggunakan antrian Amazon SQS standar sebagai antrian huruf mati. EventBridge mengirimkan peristiwa yang cocok dengan aturan ini ke antrian huruf mati jika tidak berhasil dikirim ke target. Lakukan salah satu dari berikut:

   - Pilih Tidak ada untuk tidak menggunakan antrean surat mati.

- Pilih Pilihan antrean Amazon SQS di akun AWS saat ini untuk digunakan sebagai antrean surat mati kemudian pilih antrean yang akan digunakan dari daftar menurun.
- Pilih Pilihan antrean Amazon SQS di akun AWS lainnya sebagai antrean surat mati dan kemudian masukkan ARN antrean untuk menggunakannya. Anda harus melampirkan kebijakan berbasis sumber daya ke antrean yang memberikan izin EventBridge untuk mengirim pesan padanya.

  Untuk informasi selengkapnya, lihat Memberikan izin untuk antrean surat mati.

API destination

1. (Opsional) Untuk Konfigurasi input target, pilih bagaimana Anda ingin menyesuaikan teks yang dikirim ke target untuk acara yang cocok. Pilih salah satu dari berikut:
   - Peristiwa yang cocok - EventBridge mengirimkan seluruh acara sumber asli ke target. Ini adalah default.
   - Bagian dari peristiwa yang cocok - EventBridge hanya mengirimkan bagian tertentu dari peristiwa sumber asli ke target.

     Di bawah Tentukan bagian dari peristiwa yang cocok, tentukan jalur JSON yang menentukan bagian acara yang ingin Anda kirim EventBridge ke target.
   - Konstan (teks JSON) - hanya EventBridge mengirimkan teks JSON yang ditentukan ke target. Tidak ada bagian dari acara sumber asli yang dikirim.

     Di bawah Tentukan konstanta di JSON, tentukan teks JSON yang EventBridge ingin Anda kirim ke target alih-alih acara.
   - Transformator input - Konfigurasikan transformator input untuk menyesuaikan teks yang ingin Anda EventBridge kirim ke target. Untuk informasi selengkapnya, lihat ???.

     a. Pilih Konfigurasikan transformator input.

     b. Konfigurasikan transformator input mengikuti langkah-langkah masuk???.
2. (Opsional) Di bawah kebijakan Coba lagi, tentukan cara EventBridge mencoba lagi mengirim peristiwa ke target setelah terjadi kesalahan.
   - Usia maksimum acara - Masukkan jumlah waktu maksimum (dalam jam, menit, dan detik) EventBridge untuk mempertahankan acara yang belum diproses. Defaultnya adalah 24 jam.

- Coba lagi — Masukkan jumlah maksimum kali EventBridge harus mencoba mengirim peristiwa ke target setelah terjadi kesalahan. Defaultnya adalah 185 kali.

3. (Opsional) Untuk antrian Dead-letter, pilih apakah akan menggunakan antrian Amazon SQS standar sebagai antrian huruf mati. EventBridge mengirimkan peristiwa yang cocok dengan aturan ini ke antrian huruf mati jika tidak berhasil dikirim ke target. Lakukan salah satu dari berikut:
   - Pilih Tidak ada untuk tidak menggunakan antrean surat mati.
   - Pilih Pilihan antrean Amazon SQS di akun AWS saat ini untuk digunakan sebagai antrean surat mati kemudian pilih antrean yang akan digunakan dari daftar menurun.
   - Pilih Pilihan antrean Amazon SQS di akun AWS lainnya sebagai antrean surat mati dan kemudian masukkan ARN antrean untuk menggunakannya. Anda harus melampirkan kebijakan berbasis sumber daya ke antrian yang memberikan EventBridge izin untuk mengirim pesan ke sana.

     Untuk informasi selengkapnya, lihat Memberikan izin untuk antrean surat mati.

AWS service

Perhatikan bahwa EventBridge mungkin tidak menampilkan semua bidang berikut untuk AWS layanan tertentu.

1. (Opsional) Untuk Konfigurasi input target, pilih bagaimana Anda ingin menyesuaikan teks yang dikirim ke target untuk acara yang cocok. Pilih salah satu dari berikut:
   - Peristiwa yang cocok - EventBridge mengirimkan seluruh acara sumber asli ke target. Ini adalah default.
   - Bagian dari peristiwa yang cocok - EventBridge hanya mengirimkan bagian tertentu dari peristiwa sumber asli ke target.

     Di bawah Tentukan bagian dari peristiwa yang cocok, tentukan jalur JSON yang menentukan bagian acara yang ingin Anda kirim EventBridge ke target.
   - Konstan (teks JSON) - hanya EventBridge mengirimkan teks JSON yang ditentukan ke target. Tidak ada bagian dari acara sumber asli yang dikirim.

     Di bawah Tentukan konstanta di JSON, tentukan teks JSON yang EventBridge ingin Anda kirim ke target alih-alih acara.

- Transformator input - Konfigurasikan transformator input untuk menyesuaikan teks yang ingin Anda EventBridge kirim ke target. Untuk informasi selengkapnya, lihat ???.

  a. Pilih Konfigurasikan transformator input.

  b. Konfigurasikan transformator input mengikuti langkah-langkah masuk???.

2. (Opsional) Di bawah kebijakan Coba lagi, tentukan cara EventBridge mencoba lagi mengirim peristiwa ke target setelah terjadi kesalahan.
   - Usia maksimum acara - Masukkan jumlah waktu maksimum (dalam jam, menit, dan detik) EventBridge untuk mempertahankan acara yang belum diproses. Defaultnya adalah 24 jam.
   - Coba lagi — Masukkan jumlah maksimum kali EventBridge harus mencoba mengirim peristiwa ke target setelah terjadi kesalahan. Defaultnya adalah 185 kali.
3. (Opsional) Untuk antrian Dead-letter, pilih apakah akan menggunakan antrian Amazon SQS standar sebagai antrian huruf mati. EventBridge mengirimkan peristiwa yang cocok dengan aturan ini ke antrian huruf mati jika tidak berhasil dikirim ke target. Lakukan salah satu dari berikut:
   - Pilih Tidak ada untuk tidak menggunakan antrean surat mati.
   - Pilih Pilihan antrean Amazon SQS di akun AWS saat ini untuk digunakan sebagai antrean surat mati kemudian pilih antrean yang akan digunakan dari daftar menurun.
   - Pilih Pilihan antrean Amazon SQS di akun AWS lainnya sebagai antrean surat mati dan kemudian masukkan ARN antrean untuk menggunakannya. Anda harus melampirkan kebijakan berbasis sumber daya ke antrian yang memberikan EventBridge izin untuk mengirim pesan ke sana.

     Untuk informasi selengkapnya, lihat Memberikan izin untuk antrean surat mati.
4. (Opsional) Pilih Tambahkan target lain untuk menambahkan target lain untuk aturan ini.
5. Pilih Selanjutnya.

Konfigurasikan tag dan aturan ulasan

Terakhir, masukkan tag yang diinginkan untuk aturan tersebut, lalu tinjau dan buat aturannya.

Untuk mengkonfigurasi tag, dan meninjau dan membuat aturan

1. (Opsional) Masukkan satu atau lebih tanda untuk aturan. Untuk informasi selengkapnya, lihat EventBridge Tag Amazon.

2. Pilih Next (Berikutnya).
3. Tinjau detail untuk aturan baru. Untuk membuat perubahan pada bagian mana pun, pilih tombol Edit di sebelah bagian itu.

   Saat puas dengan detail aturan, pilih Buat aturan.

Referensi ekspresi cron

Ekspresi cron memiliki enam bidang yang diperlukan, yang dipisahkan oleh spasi putih.

Sintaks

```
cron(fields)
```

Bidang	Nilai	Wildcard
Menit	0-59	, - * /
Jam	0-23	, - * /
D ay-of-month	1-31	, - * ? / L W
Bulan	1-12 atau JAN-DES	, - * /
D ay-of-week	1-7 atau MGG-SBT	, - * ? L #
Tahun	1970-2199	, - * /

Wildcard

- Wildcard , (koma) mencakup nilai tambahan. Di bidang Bulan, JAN, FEB, MAR mencakup Januari, Februari, dan Maret.
- Wildcard - (tanda hubung) menentukan rentang. Di bidang Tanggal, 1-15 mencakup tanggal 1 hingga 15 pada bulan yang ditentukan.
- Wildcard * (bintang) mencakup semua nilai di bidang. Di bidang Jam, * mencakup setiap jam. Anda tidak dapat menggunakan* di ay-of-week bidang D ay-of-month dan D. Jika Anda menggunakannya di satu bidang, Anda harus menggunakan ? di bidang lain.

- Wildcard / (garis miring) menentukan kenaikan. Di bidang menit, Anda bisa memasukkan 1/10 untuk menentukan setiap menit kesepuluh, mulai dari menit pertama jam (sebagai contoh, menit ke-11, 21, dan 31, dan seterusnya).
- Wildcard ? (tanda tanya) menentukan pilihan apa pun. Di ay-of-month bidang D Anda bisa memasukkan 7 dan jika ada hari dalam seminggu yang dapat diterima, Anda bisa masuk? di ay-of-week bidang D.
- Wildcard L di ay-of-week bidang D ay-of-month atau D menentukan hari terakhir bulan atau minggu.
- **W**Wildcard di ay-of-month bidang D menentukan hari kerja. Di ay-of-month bidang D, **3W** tentukan hari kerja yang paling dekat dengan hari ketiga bulan itu.
- Wildcard # di ay-of-week bidang D menentukan contoh tertentu dari hari yang ditentukan dalam seminggu dalam sebulan. Sebagai contoh, 3#2 akan menjadi hari Selasa kedua setiap bulan: 3 mengacu pada hari Selasa karena itu adalah hari ketiga setiap minggu, dan 2 mengacu pada hari kedua dari jenis tersebut dalam bulan tersebut.

> **Note**
>
> Jika Anda menggunakan karakter '#', Anda hanya dapat menentukan satu ekspresi di day-of-week bidang. Sebagai contoh, `"3#1,6#3"` tidak valid karena ditafsirkan sebagai dua ekspresi.

Keterbatasan:

- Anda tidak dapat menentukan ay-of-week bidang D ay-of-month dan D dalam ekspresi cron yang sama. Jika Anda menentukan nilai atau * (bintang) di salah satu bidang, Anda harus menggunakan ? (tanda tanya) di bidang lain.
- Ekspresi cron yang mengarah ke tingkat lebih cepat dari 1 menit tidak didukung.

Contoh

Anda dapat menggunakan contoh string cron berikut saat membuat aturan dengan jadwal.

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
0	10	*	*	?	*	Jalankan pada pukul 10.00 (UTC+0) setiap hari
15	12	*	*	?	*	Jalankan pada pukul 12.15 (UTC+0) setiap hari
0	18	?	*	SNN-JMT	*	Jalankan pada pukul 18.00 (UTC) setiap Senin hingga Jumat
0	8	1	*	?	*	Jalankan pada pukul 08.00 (UTC+0) setiap tanggal 1 pada bulan tersebut
0/15	*	*	*	?	*	Jalankan setiap 15 menit

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
0/10	*	?	*	MON-FRI	*	Jalankan setiap 10 menit Senin hingga Jumat
0/5	8-17	?	*	SNN-JMT	*	Jalankan setiap 5 menit pada hari Senin hingga Jumat antara pukul 08.00 dan 17.55 (UTC+0)

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
0/30	20-2	?	*	SNN-JMT	*	Jalankan setiap 30 menit pada hari Senin sampai Jumat antara pukul 22.00 pada hari pertama sampai 02.00 pada hari berikutnya (UTC) Jalankan dari pukul 12:00 hingga 2:00 pagi pada Senin pagi (UTC).

Contoh berikut membuat aturan yang berjalan setiap hari pada pukul 12.00 siang UTC+0.

```
aws events put-rule --schedule-expression "cron(0 12 * * ? *)" --name MyRule1
```

Contoh berikut membuat aturan yang berjalan setiap hari, pada 14.05 dan 14.35 UTC+0.

```
aws events put-rule --schedule-expression "cron(5,35 14 * * ? *)" --name MyRule2
```

Contoh berikut membuat aturan yang berjalan pada 10.15 UTC+0 pada hari Jumat terakhir setiap bulan selama tahun 2019 hingga 2022.

```
aws events put-rule --schedule-expression "cron(15 10 ? * 6L 2019-2022)" --name MyRule3
```

Referensi ekspresi nilai

Ekspresi tingkat dimulai ketika Anda membuat aturan acara terjadwal, dan kemudian aturan berjalan pada jadwal yang ditetapkan.

Ekspresi tingkat memiliki dua bidang yang diperlukan, yang dipisahkan oleh spasi putih.

Sintaks

```
rate(value unit)
```

nilai

Bilangan positif

unit

Unit waktu. Unit yang berbeda diperlukan untuk nilai 1, seperti `minute`, dan nilai lebih dari 1, seperti `minutes`.

Nilai yang valid: menit | menit-menit | jam | jam-jam | hari | hari-hari

Keterbatasan:

Jika nilai sama dengan 1, maka unit harus tunggal. Jika nilai lebih besar dari 1, unit harus jamak. Misalnya, tingkat(1 jam-jam) dan tarif(5 jam) tidak valid, tetapi tarif (1 jam) dan tarif(5 jam-jam) berlaku.

Contoh

Contoh-contoh berikut menunjukkan cara menggunakan ekspresi tingkat dengan perintah AWS CLI `put-rule`. Contoh pertama memicu aturan setiap menit, yang berikutnya memicunya setiap lima menit, contoh ketiga memicunya sekali dalam satu jam, dan contoh terakhir memicunya sekali sehari.

```
aws events put-rule --schedule-expression "rate(1 minute)" --name MyRule2
```

```
aws events put-rule --schedule-expression "rate(5 minutes)" --name MyRule3
```

```
aws events put-rule --schedule-expression "rate(1 hour)" --name MyRule4
```

```
aws events put-rule --schedule-expression "rate(1 day)" --name MyRule5
```

Menonaktifkan atau menghapus aturan Amazon EventBridge

Untuk menghentikan aturan dari mengolah peristiwa atau berjalan pada jadwal, Anda dapat menghapus atau menonaktifkan aturan. Langkah-langkah berikut akan memandu Anda dalam menghapus atau menonaktifkan aturan EventBridge.

Menghapus atau menonaktifkan aturan

1. Buka konsol Amazon EventBridge di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.

   Di bawah Bus peristiwa, pilih bus peristiwa yang terkait dengan aturan.
3. Lakukan salah satu dari berikut ini:

   a. Untuk menghapus aturan, pilih tombol di samping aturan dan pilih Tindakan, Hapus, Hapus.

      Jika aturan adalah aturan terkelola, masukkan nama aturan untuk mengakui bahwa aturan itu adalah aturan terkelola dan bahwa menghapusnya dapat menghentikan fungsionalitas dalam layanan yang membuat aturan. Untuk melanjutkan, masukkan nama aturan dan pilih Hapus paksa.

   b. Untuk menonaktifkan aturan sementara, pilih tombol di samping aturan dan pilih Nonaktifkan, Nonaktifkan.

      Anda tidak dapat menonaktifkan aturan terkelola.

Praktik terbaik saat mendefinisikan aturan Amazon EventBridge

Berikut adalah beberapa praktik terbaik yang perlu dipertimbangkan saat Anda membuat aturan untuk bus acara Anda.

Tetapkan satu target untuk setiap aturan

Meskipun Anda dapat menentukan hingga lima target untuk aturan tertentu, mengelola aturan akan lebih mudah bila Anda menentukan satu target untuk setiap aturan. Jika lebih dari satu target perlu menerima kumpulan peristiwa yang sama, sebaiknya duplikasi aturan untuk mengirimkan peristiwa yang sama ke target yang berbeda. Enkapsulasi ini menyederhanakan pemeliharaan aturan: jika kebutuhan target peristiwa menyimpang dari waktu ke waktu, Anda dapat memperbarui setiap aturan dan pola kejadiannya secara independen dari yang lain.

Tetapkan izin aturan

Anda dapat mengaktifkan komponen atau layanan aplikasi yang memakan acara untuk mengendalikan pengelolaan aturan mereka sendiri. Pendekatan arsitektur umum yang diadopsi oleh pelanggan adalah mengisolasi komponen atau layanan aplikasi ini dengan menggunakan AWS akun terpisah. Untuk mengaktifkan aliran peristiwa dari satu akun ke akun lain, Anda harus membuat aturan pada satu bus peristiwa yang merutekan acara ke bus acara di akun lain. Anda dapat mengaktifkan tim atau layanan yang memakan acara untuk mengendalikan pengelolaan aturan mereka sendiri. Anda melakukannya dengan menentukan izin yang sesuai ke akun mereka melalui kebijakan sumber daya. Ini berfungsi di seluruh akun dan Wilayah.

Untuk informasi selengkapnya, lihat ???.

Misalnya kebijakan sumber daya, lihat Pola desain multi-akun dengan EventBridge Amazon GitHub aktif.

Pantau kinerja aturan

Pantau aturan Anda untuk memastikan performa seperti yang Anda harapkan:

- Memantau `TriggeredRules` metrik untuk kehilangan data-point atau anomali dapat membantu Anda dalam mendeteksi perbedaan untuk penerbit yang membuat perubahan melanggar. Untuk informasi selengkapnya, lihat ???.
- Alarm pada anomali atau jumlah maksimum yang diharapkan juga dapat membantu mendeteksi ketika aturan cocok dengan peristiwa baru. Hal ini dapat terjadi ketika penerbit acara, termasuk AWS layanan dan mitra SaaS, memperkenalkan acara baru saat mengaktifkan kasus penggunaan dan fitur baru. Ketika peristiwa baru ini tidak terduga dan mengarah ke volume yang lebih tinggi daripada tingkat pemrosesan target hilir, mereka dapat menyebabkan backlog peristiwa.

  Pemrosesan kejadian tak terduga semacam itu juga dapat menyebabkan biaya penagihan yang tidak diinginkan.

  Hal ini juga dapat memicu pembatasan aturan ketika akun melewati pemanggilan target agregatnya per kuota layanan kedua. EventBridgemasih akan mencoba untuk mengirimkan peristiwa yang cocok dengan aturan yang dibatasi dan mencoba kembali hingga 24 jam atau seperti yang dijelaskan dalam kebijakan percobaan ulang kustom target. Anda dapat mendeteksi dan alarm aturan throttled menggunakan metrik `ThrottledRules`
- Untuk kasus penggunaan latensi rendah, Anda juga dapat memantau penggunaan latensi`IngestionToInvocationStartLatency`, yang memberikan indikasi kesehatan

bus acara Anda. Setiap periode latensi tinggi yang diperpanjang selama 30 detik dapat mengindikasikan gangguan layanan atau pembatasan aturan.

Menggunakan Amazon EventBridge danAWS Serverless Application Modeltemplat

Anda dapat membangun dan mengujiaturansecara manual di konsol EventBridge, yang dapat membantu dalam proses pengembangan saat Anda menyempurnakanpola peristiwa. Namun, setelah Anda siap untuk menyebarkan aplikasi Anda, lebih mudah untuk menggunakan kerangka kerja sepertiAWS SAMuntuk meluncurkan semua sumber daya tanpa server Anda secara konsisten.

Kita akan menggunakan inicontoh aplikasiuntuk melihat ke dalam cara yang dapat Anda gunakanAWS SAMtemplate untuk membangun sumber daya EventBridge. File template.yaml dalam contoh ini adalahAWS SAMtemplate yang mendefinisikan empatAWS Lambdafungsi dan menunjukkan dua cara berbeda untuk mengintegrasikan fungsi Lambda dengan EventBridge.

Untuk panduan dari contoh aplikasi ini, lihat???.

Ada dua pendekatan untuk menggunakan EventBridge danAWS SAMtemplat. Untuk integrasi sederhana di mana satu fungsi Lambda dipanggil oleh satu aturan,Templat gabungandianjurkan. Jika Anda memiliki logika routing yang kompleks, atau Anda terhubung ke sumber daya di luar AndaAWS SAMtemplatTemplat terpisahadalah pilihan yang lebih baik.

Pendekatan:

- Templat gabungan
- Templat terpisah

Templat gabungan

Pendekatan pertama menggunakan`Events`properti untuk mengkonfigurasi aturan EventBridge. Contoh kode berikut mendefinisikan sebuahperistiwayang memanggil fungsi Lambda Anda.

> **Note**
>
> Contoh ini secara otomatis membuat aturan pada defaultBus peristiwa, yang ada di setiapAWSakun. Untuk mengaitkan aturan dengan bus acara khusus, Anda dapat menambahkan`EventBusName`ke templat.

```
atmConsumerCase3Fn:
  Type: AWS::Serverless::Function
```

```
  Properties:
    CodeUri: atmConsumer/
    Handler: handler.case3Handler
    Runtime: nodejs12.x
    Events:
      Trigger:
        Type: CloudWatchEvent
        Properties:
          Pattern:
            source:
              - custom.myATMapp
            detail-type:
              - transaction
            detail:
              result:
                - "anything-but": "approved"
```

Kode YAML ini setara dengan pola acara di konsol EventBridge. Di YAML, Anda hanya perlu menentukan pola acara, danAWS SAMBuat peran IAM secara otomatis dengan izin yang diperlukan.

Templat terpisah

Dalam pendekatan kedua untuk mendefinisikan konfigurasi EventBridge diAWS SAM, sumber daya dipisahkan lebih jelas dalam template.

1. Pertama, Anda mendefinisikan fungsi Lambda:

   ```
   atmConsumerCase1Fn:
     Type: AWS::Serverless::Function
     Properties:
       CodeUri: atmConsumer/
       Handler: handler.case1Handler
       Runtime: nodejs12.x
   ```

2. Selanjutnya, tentukan aturan menggunakan`AWS::Events::Rule`sumber daya. Properti mendefinisikan pola acara dan juga dapat menentukansasaran. Anda dapat secara eksplisit menentukan beberapa target.

   ```
   EventRuleCase1:
     Type: AWS::Events::Rule
     Properties:
       Description: "Approved transactions"
   ```

```
    EventPattern:
      source:
        - "custom.myATMapp"
      detail-type:
        - transaction
      detail:
        result:
          - "approved"
    State: "ENABLED"
    Targets:
      -
        Arn:
          Fn::GetAtt:
            - "atmConsumerCase1Fn"
            - "Arn"
        Id: "atmConsumerTarget1"
```

3. Akhirnya, mendefinisikan`AWS::Lambda::Permission`sumber daya yang memberikan izin untuk EventBridge untuk memanggil target.

```
PermissionForEventsToInvokeLambda:
  Type: AWS::Lambda::Permission
  Properties:
    FunctionName:
      Ref: "atmConsumerCase1Fn"
    Action: "lambda:InvokeFunction"
    Principal: "events.amazonaws.com"
    SourceArn:
      Fn::GetAtt:
        - "EventRuleCase1"
        - "Arn"
```

Buat AWS CloudFormation template dari EventBridge aturan Amazon

AWS CloudFormationmemungkinkan Anda mengonfigurasi dan mengelola AWS sumber daya Anda di seluruh akun dan wilayah secara terpusat dan berulang dengan memperlakukan infrastruktur sebagai kode. CloudFormation melakukan ini dengan membiarkan Anda membuat template, yang menentukan sumber daya yang ingin Anda sediakan dan kelola.

EventBridge memungkinkan Anda untuk membuat template dari aturan yang ada di akun Anda, sebagai bantuan untuk membantu Anda mulai mengembangkan CloudFormation template. Anda dapat memilih satu aturan, atau beberapa aturan untuk disertakan dalam template. Anda kemudian dapat menggunakan template ini sebagai dasar untuk membuat tumpukan sumber daya yang CloudFormation dikelola.

Untuk informasi lebih lanjut tentang CloudFormation lihat Panduan AWS CloudFormation Pengguna.

> **Note**
>
> EventBridge tidak menyertakan aturan terkelola dalam template yang dihasilkan.

Anda juga dapat membuat template dari bus acara yang ada, termasuk aturan yang berisi bus acara.

Untuk menghasilkan AWS CloudFormation template dari satu atau beberapa aturan

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Di bawah Pilih bus acara, pilih bus acara yang berisi aturan yang ingin Anda sertakan dalam templat.
4. Di bawah Aturan, pilih aturan yang ingin Anda sertakan dalam AWS CloudFormation templat yang dihasilkan.

   Untuk satu aturan, Anda juga dapat memilih nama aturan untuk menampilkan halaman detail aturan.
5. Pilih CloudFormation Template, lalu pilih format mana yang EventBridge ingin Anda buat template di: JSON atau YAMG.

   EventBridge menampilkan template, yang dihasilkan dalam format yang dipilih.
6. EventBridge memberi Anda pilihan untuk mengunduh file template, atau menyalin template ke clipboard.

   - Untuk mengunduh file templat, pilih Unduh.
   - Untuk menyalin template ke clipboard, pilih Salin.
7. Untuk keluar dari template, pilih Batal.

Setelah Anda menyesuaikan AWS CloudFormation template Anda seperlunya untuk kasus penggunaan Anda, Anda dapat menggunakannya untuk membuat tumpukan. AWS CloudFormation

Pertimbangan saat menggunakan CloudFormation template yang dihasilkan dari Amazon EventBridge

Pertimbangkan faktor-faktor berikut saat menggunakan CloudFormation templat yang Anda hasilkan EventBridge:

- EventBridge tidak menyertakan kata sandi apa pun dalam template generate.

  Anda dapat mengedit templat untuk menyertakan parameter templat yang memungkinkan pengguna menentukan kata sandi atau informasi sensitif lainnya saat menggunakan templat untuk membuat atau memperbarui CloudFormation tumpukan.

  Selain itu, pengguna dapat menggunakan Secrets Manager untuk membuat rahasia di wilayah yang diinginkan dan kemudian mengedit template yang dihasilkan untuk menggunakan parameter dinamis.
- Target dalam template yang dihasilkan tetap persis seperti yang ditentukan dalam bus acara asli. Hal ini dapat menyebabkan masalah lintas wilayah jika Anda tidak mengedit template dengan tepat sebelum menggunakannya untuk membuat tumpukan di wilayah lain.

  Selain itu, template yang dihasilkan tidak membuat target hilir secara otomatis.

EventBridge Target Amazon

Target adalah sumber daya atau titik akhir yang EventBridge mengirimkan peristiwa ke saat acara cocok dengan pola acara yang ditentukan untuk aturan. Aturan memproses data peristiwa dan mengirimkan informasi yang berkaitan ke target. Untuk mengirimkan data peristiwa ke target, EventBridge perlu izin untuk mengakses sumber daya target. Anda dapat menentukan hingga lima target untuk setiap aturan.

Ketika Anda menambahkan target ke aturan dan aturan tersebut berjalan segera setelah itu, setiap target baru atau diperbarui mungkin tidak segera dipanggil. Berikan waktu yang singkat agar perubahan diterapkan.

Video berikut mencakup dasar-dasar target: Apa itu target

Target tersedia di EventBridge konsol

Anda dapat mengonfigurasi target berikut untuk acara di EventBridge konsol:

- Tujuan API
- API Gateway
- AWS AppSync;
- Antrian pekerjaan batch
- CloudWatch grup log
- CodeBuild proyek
- CodePipeline
- Panggilan `CreateSnapshot` API Amazon EBS
- EC2 Image Builder
- Panggilan `RebootInstances` API EC2
- Panggilan `StopInstances` API EC2
- Panggilan `TerminateInstances` API EC2
- Tugas ECS
- Bus acara di akun atau Wilayah yang berbeda
- Bus acara di akun dan Wilayah yang sama

- Aliran pengiriman Firehose
- Glue alur kerja
- Rencana respons Manajer Insiden
- Templat penilaian Inspector
- Aliran kinesis
- Fungsi Lambda (ASYNC)
- Kueri API data klaster Amazon Redshift
- Kueri API data grup kerja Amazon Redshift Tanpa Server
- SageMaker Pipa
- Topik Amazon SNS

  EventBridge tidak mendukung topik Amazon SNS FIFO (masuk pertama, keluar pertama).
- Antrean Amazon SQS
- Mesin status Step Functions (ASYNC)
- Otomatisasi Systems Manager
- Systems Manager OpsItem
- Run Command Systems Manager

Parameter terget

Beberapa target tidak mengirim informasi dalam muatan acara ke target, sebaliknya, mereka memperlakukan peristiwa sebagai pemicu untuk menjalankan API tertentu. EventBridge menggunakan parameter Target untuk menentukan apa yang terjadi dengan target itu. Sumber daya yang dimaksud meliputi:

- Tujuan API (Data yang dikirim ke tujuan API harus sesuai dengan struktur API. Anda harus menggunakan `InputTransformer`objek untuk memastikan data terstruktur dengan benar. Jika Anda ingin memasukkan muatan acara asli, rujuk di `InputTransformer`.)
- API Gateway (Data yang dikirim ke API Gateway harus sesuai dengan struktur API. Anda harus menggunakan `InputTransformer`objek untuk memastikan data terstruktur dengan benar. Jika Anda ingin memasukkan muatan acara asli, rujuk di `InputTransformer`.)
- EC2 Image Builder Amazon
- `RedshiftDataParameters`(Cluster API Data Amazon Redshift)

- `SageMakerPipelineParameters`(Pipa Pembuatan Model SageMaker Runtime Amazon)

> **Note**
>
> EventBridge tidak mendukung semua sintaks JSON Path dan mengevaluasinya saat runtime. Sintaks yang didukung meliputi:
>
> - notasi titik (misalnya,`$.detail`)
> - tanda hubung
> - menggarisbawahi
> - karakter alfanumerik
> - indeks array
> - wildcard (*)

Parameter jalur dinamis

Beberapa parameter target mendukung sintaks jalur JSON dinamis opsional. Sintaks ini memungkinkan Anda untuk menentukan jalur JSON bukan nilai statis (misalnya`$.detail.state`). Seluruh nilai harus berupa jalur JSON, bukan hanya bagian darinya. Misalnya, `RedshiftParameters.Sql` bisa `$.detail.state` tetapi tidak bisa`"SELECT * FROM $.detail.state"`. Jalur ini diganti secara dinamis saat runtime dengan data dari payload acara itu sendiri di jalur yang ditentukan. Parameter jalur dinamis tidak dapat mereferensikan nilai baru atau yang diubah yang dihasilkan dari transformasi input. Sintaks yang didukung untuk jalur JSON parameter dinamis sama seperti saat mengubah input. Lihat informasi yang lebih lengkap di ???

Sintaks dinamis dapat digunakan pada semua string, bidang non-enum dari parameter ini:

- `EcsParameters`
- `HttpParameters`(kecuali `HeaderParameters` kunci)
- `RedshiftDataParameters`
- `SageMakerPipelineParameters`

Izin

Untuk melakukan panggilan API pada sumber daya yang Anda miliki, EventBridge perlu izin yang sesuai. Untuk AWS Lambda dan sumber daya Amazon SNS, EventBridge menggunakan kebijakan berbasis sumber daya. Untuk instans EC2, aliran data Kinesis, dan mesin status Step Functions, EventBridge menggunakan peran IAM yang Anda tentukan dalam parameter. `RoleARN` `PutTargets` Anda dapat memanggil titik akhir API Gateway dengan otorisasi IAM yang dikonfigurasi, tetapi peran tersebut bersifat opsional jika Anda belum mengonfigurasi otorisasi. Untuk informasi selengkapnya, lihat Amazon EventBridge dan AWS Identity and Access Management.

Jika akun lainnya berada di Wilayah yang sama dan telah memberikan izin kepada Anda, maka Anda dapat mengirim peristiwa ke akun tersebut. Untuk informasi selengkapnya, lihat Mengirim dan menerima EventBridge acara Amazon antar AWS akun.

Jika target Anda dienkripsi, Anda harus menyertakan bagian berikut dalam kebijakan kunci KMS Anda.

```
{
    "Sid": "Allow EventBridge to use the key",
    "Effect": "Allow",
    "Principal": {
        "Service": "events.amazonaws.com"
    },
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
    ],
    "Resource": "*"
}
```

EventBridge spesifik target

AWS Batch antrian pekerjaan

Parameter tertentu AWS Batch `submitJob` dapat dikonfigurasi melalui BatchParameters.

Lainnya dapat ditentukan dalam muatan acara. Jika payload peristiwa (melewati atau melalui InputTransformers) berisi kunci berikut, mereka dipetakan untuk `submitJob` meminta parameter:

- `ContainerOverrides: containerOverrides`

> **Note**
>
> Ini hanya mencakup perintah, lingkungan, memori, dan vcpus

- `DependsOn: dependsOn`

> **Note**
>
> Ini termasuk hanya JoBid

- `Parameters: parameters`

CloudWatch Grup log

Jika Anda tidak menggunakan target InputTransformerwith a CloudWatch Logs, payload peristiwa akan digunakan sebagai pesan log, dan sumber acara sebagai stempel waktu. Jika Anda menggunakan InputTransformer, template harus:

```
{"timestamp":<timestamp>,"message":<message>}
```

EventBridge batch entri yang dikirim ke aliran log; oleh karena itu, EventBridge dapat mengirimkan satu atau beberapa peristiwa ke aliran log, tergantung pada lalu lintas.

CodeBuild proyek

Jika Anda menggunakan InputTransformersuntuk membentuk peristiwa input ke Target agar sesuai dengan CodeBuild StartBuildRequeststruktur, parameter akan dipetakan 1-ke-1 dan diteruskan ke. `codeBuild.StartBuild`

Tugas Amazon ECS

Jika Anda menggunakan InputTransformersuntuk membentuk peristiwa input ke Target agar sesuai dengan RunTask TaskOverridestruktur Amazon ECS, parameter akan dipetakan 1-ke-1 dan diteruskan ke. `ecs.RunTask`

Rencana Respons Manajer Insiden

Jika peristiwa yang cocok berasal dari CloudWatch Alarm, detail perubahan status alarm diisi ke detail pemicu StartIncidentRequest panggilan ke Manajer Insiden.

Konfigurasi target

Pelajari cara mengonfigurasi pengaturan untuk EventBridge target.

Target:

- Tujuan API
- EventBridge Target Amazon untuk Amazon API Gateway
- AWS AppSync target untuk Amazon EventBridge
- Koneksi untuk target titik akhir HTTP
- Mengirim dan menerima EventBridge acara Amazon antar AWS akun
- Mengirim dan menerima EventBridge acara Amazon antar AWS Wilayah
- Mengirim dan menerima EventBridge acara Amazon antara bus acara di akun dan Wilayah yang sama

Tujuan API

Tujuan Amazon EventBridge API adalah titik akhir HTTP yang dapat Anda panggil sebagai target aturan, mirip dengan cara Anda memanggil AWS layanan atau sumber daya sebagai target. Menggunakan tujuan API, Anda dapat merutekan peristiwa antara AWS layanan, aplikasi perangkat lunak terintegrasi sebagai layanan (SaaS), dan aplikasi Anda di luar AWS dengan menggunakan panggilan API. Saat Anda menentukan tujuan API sebagai target aturan, EventBridge memanggil titik akhir HTTP untuk setiap peristiwa yang cocok dengan pola peristiwa yang ditentukan dalam aturan, lalu mengirimkan informasi peristiwa dengan permintaan tersebut. Dengan EventBridge, Anda dapat menggunakan metode HTTP apa pun kecuali CONNECT dan TRACE untuk permintaan tersebut. Metode HTTP yang paling umum digunakan adalah PUT dan POST. Anda juga dapat menggunakan pengubah masukan untuk menyesuaikan peristiwa dengan parameter pada parameter titik akhir HTTP spesifik. Untuk informasi selengkapnya, lihat Transformasi EventBridge masukan Amazon.

Tujuan API tidak mendukung tujuan pribadi, seperti titik akhir VPC antarmuka. Untuk informasi selengkapnya, lihat ???.

> **Important**
>
> EventBridge permintaan ke titik akhir tujuan API harus memiliki batas waktu eksekusi klien maksimum 5 detik. Jika titik akhir target membutuhkan waktu lebih dari 5 detik untuk merespons, EventBridge hentikan permintaan. EventBridge percobaan ulang permintaan habis waktu hingga maksimum yang dikonfigurasi pada kebijakan coba ulang Anda. Secara default maksimal adalah 24 jam dan 185 kali. Setelah jumlah maksimum coba lagi, peristiwa dikirim ke antrean surat mati Anda jika Anda memilikinya. Jika tidak, peristiwa digagalkan.

Video berikut menunjukkan penggunaan tujuan API: Menggunakan tujuan API

Dalam topik ini:

- Buat tujuan API
- Membuat aturan yang mengirim peristiwa ke tujuan API
- Peran tertaut layanan untuk tujuan API
- Header dalam permintaan ke tujuan API
- Kode kesalahan tujuan API

- Bagaimana tingkat invokasi memengaruhi pengiriman peristiwa
- Mengirim CloudEvents acara ke tujuan API
- Mitra tujuan API

Buat tujuan API

Setiap tujuan API membutuhkan koneksi. Koneksi menentukan jenis otorisasi dan kredensi yang akan digunakan untuk mengotorisasi dengan titik akhir tujuan API. Anda dapat memilih koneksi yang ada, atau membuat koneksi pada saat yang sama saat Anda membuat tujuan API. Lihat informasi yang lebih lengkap di ???

Untuk membuat tujuan API menggunakan EventBridge konsol

1. Masuk untuk AWS menggunakan akun yang memiliki izin untuk mengelola EventBridge dan membuka EventBridgekonsol.
2. Di panel navigasi kiri, pilih Tujuan API.
3. Gulir ke bawah ke tabel Tujuan API, dan kemudian pilih Buat tujuan API.
4. Pada halaman Buat tujuan API, masukkan Nama untuk tujuan API. Anda dapat menggunakan hingga 64 karakter huruf besar atau huruf kecil, angka, titik (.), tanda hubung (-), atau garis bawah (_).

   Nama harus unik untuk akun Anda di Wilayah saat ini.
5. Masukkan Deskripsi untuk tujuan API.
6. Masukkan Titik akhir tujuan API untuk tujuan API. Titik akhir tujuan API adalah target titik akhir invokasi HTTP untuk peristiwa. Informasi otorisasi yang Anda sertakan dalam koneksi yang digunakan untuk tujuan API ini digunakan untuk mengotorisasi terhadap titik akhir ini. URL harus menggunakan HTTPS.
7. Masukkan Metode HTTP untuk digunakan untuk menyambungkan ke Titik akhir tujuan API.
8. (Opsional) Untuk bidang Batas tingkat permintaan per detik, masukkan jumlah maksimum invokasi per detik untuk mengirim ke titik akhir tujuan API.

   Batas tarif yang Anda tetapkan dapat memengaruhi cara EventBridge menyampaikan acara. Untuk informasi selengkapnya, lihat Bagaimana tingkat invokasi memengaruhi pengiriman peristiwa.
9. Untuk Koneksi, lakukan salah satu langkah berikut ini:

- Pilih Gunakan koneksi yang ada, dan kemudian pilih koneksi yang akan digunakan untuk tujuan API ini.
- Pilih Buat koneksi baru, dan kemudian masukkan detail untuk koneksi yang akan dibuat. Untuk informasi selengkapnya, lihat Koneksi.

10. Pilih Buat.

Membuat aturan yang mengirim peristiwa ke tujuan API

Setelah Anda membuat tujuan API, Anda dapat memilihnya sebagai target aturan. Untuk menggunakan tujuan API sebagai target, Anda harus menyediakan IAM role dengan izin yang benar. Untuk informasi selengkapnya, lihat ???

Memilih tujuan API sebagai target adalah bagian dari pembuatan aturan.

Untuk membuat aturan yang mengirimkan peristiwa ke tujuan API menggunakan konsol

1. Ikuti langkah-langkah dalam ??? prosedur.
2. Pada ??? langkahnya, saat diminta untuk memilih tujuan API sebagai tipe target:

   a. Pilih tujuan EventBridge API.

   b. Lakukan salah satu hal berikut:

      - Pilih Gunakan tujuan API yang ada dan pilih tujuan API yang ada
      - Pilih Buat tujuan API baru dan tentukan setelan yang diperlukan untuk menentukan tujuan API baru Anda.

        Untuk informasi selengkapnya tentang menentukan pengaturan yang diperlukan, lihat???.

   c. (Opsional): Untuk menentukan parameter header untuk acara tersebut, di bawah Parameter Header pilih Tambahkan parameter header.

      Selanjutnya, tentukan kunci dan nilai untuk parameter header.

   d. (Opsional): Untuk menentukan parameter string kueri untuk acara tersebut, di bawah Parameter string kueri pilih Tambahkan parameter string kueri.

      Selanjutnya, tentukan kunci dan nilai untuk parameter string kueri.

3. Selesaikan pembuatan aturan mengikuti langkah-langkah prosedur.

Peran tertaut layanan untuk tujuan API

Saat Anda membuat sambungan untuk tujuan API, peran yang ditautkan layanan bernama akan AWS ServiceRoleForAmazonEventBridgeApiDestinationsditambahkan ke akun Anda. EventBridge menggunakan peran terkait layanan untuk membuat dan menyimpan rahasia di Secrets Manager. Untuk memberikan izin yang diperlukan ke peran terkait layanan, EventBridge lampirkan AmazonEventBridgeApiDestinationsServiceRolePolicykebijakan ke peran tersebut. Kebijakan membatasi izin yang diberikan hanya kepada mereka yang diperlukan bagi peran untuk berinteraksi dengan rahasia untuk koneksi. Tidak ada izin lain yang disertakan, dan peran hanya dapat berinteraksi dengan koneksi di akun Anda untuk mengelola rahasia.

Kebijakan berikut adalah`AmazonEventBridgeApiDestinationsServiceRolePolicy`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:CreateSecret",
                "secretsmanager:UpdateSecret",
                "secretsmanager:DescribeSecret",
                "secretsmanager:DeleteSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue"
            ],
            "Resource": "arn:aws:secretsmanager:*:*:secret:events!connection/*"
        }
    ]
}
```

Untuk informasi selengkapnya tentang peran tertaut layanan, lihat Menggunakan peran tertaut layanan dalam dokumentasi IAM.

Peran `AmazonEventBridgeApiDestinationsServiceRolePolicy` terkait layanan didukung di wilayah berikut: AWS

- AS Timur (N. Virginia)
- AS Timur (Ohio)
- AS Barat (California Utara)

- AS Barat (Oregon)
- Afrika (Cape Town)
- Asia Pasifik (Hong Kong)
- Asia Pasifik (Mumbai)
- Asia Pasifik (Osaka)
- Asia Pasifik (Seoul)
- Asia Pasifik (Singapura)
- Asia Pasifik (Sydney)
- Asia Pasifik (Tokyo)
- Kanada (Pusat)
- Eropa (Frankfurt)
- Eropa (Irlandia)
- Eropa (London)
- Eropa (Milan)
- Eropa (Paris)
- Eropa (Stockholm)
- Eropa (Milan)
- Amerika Selatan (Sao Paulo)
- Tiongkok (Ningxia)
- Tiongkok (Beijing)

Header dalam permintaan ke tujuan API

Bagian berikut merinci cara EventBridge menangani header HTTP dalam permintaan ke tujuan API.

Header yang disertakan dalam permintaan ke tujuan API

Selain header otorisasi yang ditentukan untuk koneksi yang digunakan untuk tujuan API, EventBridge sertakan header berikut di setiap permintaan.

Kunci header	Nilai header
User-Agent	Amazon//EventBridgeApiDestinations

Kunci header	Nilai header
Content-Type	Jika tidak ada nilai Content-Type kustom yang ditentukan, EventBridge sertakan nilai default berikut sebagai Content-Type: application/json; charset=utf-8
Kisaran	bytes=0-1048575
Terima-Pengenkodean	gzip,mengempis
Koneksi	tutup
Panjang Konten	Header entitas yang menunjukkan ukuran bodi entitas, dalam byte, yang dikirim ke penerima.
Host	Header permintaan yang menentukan host dan nomor port server di mana permintaan sedang dikirim.

Header yang tidak dapat diganti dalam permintaan ke tujuan API

EventBridge tidak memungkinkan Anda untuk mengganti header berikut:

- User-Agent
- Kisaran

Header EventBridge menghapus dari permintaan ke tujuan API

EventBridge menghapus header berikut untuk semua permintaan tujuan API:

- A-IM
- Terima-Charset
- Terima-Datetime
- Terima-Pengenkodean
- Kontrol Cache
- Koneksi

- Pengkodean Konten
- Content-Length
- Content-MD5
- Tanggal
- Harapkan
- Diteruskan
- Dari
- Host
- HTTP2-pengaturan
- Jika-Pertandingan
- Jika-Modifikasi-Sejak
- Jika-Tidak Pertandingan
- Jika-Rentang
- Jika-Tidak Dimodifikasi-Sejak
- Max-Maju
- Asal
- Pragma
- Otorisasi Proksi
- Kisaran
- Perujuk
- TE
- Trailer
- Transfer-Encoding
- User-Agent
- Peningkatan
- Melalui
- Peringatan

Kode kesalahan tujuan API

Saat EventBridge mencoba mengirimkan acara ke tujuan API dan terjadi kesalahan, EventBridge lakukan hal berikut:

- Peristiwa yang terkait dengan kode kesalahan 409, 429, dan 5xx dicoba lagi.
- Peristiwa yang berkaitan dengan kode kesalahan 1xx, 2xx, 3xx, dan 4xx (tidak termasuk 429) tidak dicoba lagi.

EventBridge Tujuan API membaca header respons HTTP standar `Retry-After` untuk mengetahui berapa lama menunggu sebelum membuat permintaan tindak lanjut. EventBridge memilih nilai yang lebih konservatif antara kebijakan coba ulang yang ditentukan dan header. `Retry-After` Jika `Retry-After` nilainya negatif, EventBridge berhenti mencoba kembali pengiriman untuk acara itu.

Bagaimana tingkat invokasi memengaruhi pengiriman peristiwa

Jika Anda menetapkan tingkat invokasi per detik untuk nilai yang jauh lebih rendah daripada jumlah invokasi yang dihasilkan, peristiwa tidak dapat dikirimkan dalam waktu 24 jam coba lagi untuk peristiwa. Sebagai contoh, jika Anda menetapkan tingkat invokasi ke 10 invokasi per detik, tetapi ribuan peristiwa per detik dihasilkan, Anda akan dengan cepat memiliki backlog peristiwa untuk mengirimkan yang melebihi 24 jam. Untuk memastikan bahwa tidak ada peristiwa yang hilang, atur antrean surat mati untuk mengirim peristiwa dengan invokasi yang gagal sehingga Anda dapat memproses peristiwa di lain waktu. Untuk informasi selengkapnya, lihat Kebijakan coba lagi peristiwa dan menggunakan antrean surat mati.

Mengirim CloudEvents acara ke tujuan API

CloudEvents adalah spesifikasi vendor netral untuk pemformatan acara, dengan tujuan menyediakan interoperabilitas di seluruh layanan, platform, dan sistem. Anda dapat menggunakan EventBridge untuk mengubah peristiwa AWS layanan CloudEvents sebelum dikirim ke target, seperti tujuan API.

> **Note**
>
> Prosedur berikut menjelaskan cara mengubah peristiwa sumber menjadi mode terstruktur CloudEvents. Dalam CloudEvents spesifikasi, pesan mode terstruktur adalah pesan di mana seluruh peristiwa (atribut dan data) dikodekan ke dalam muatan acara.

Untuk informasi selengkapnya tentang CloudEvents spesifikasi, lihat cloudevents.io.

Untuk mengubah AWS peristiwa ke CloudEvents format menggunakan konsol

Untuk mengubah acara ke CloudEvents format sebelum pengiriman ke target, Anda mulai dengan membuat aturan bus acara. Sebagai bagian dari mendefinisikan aturan, Anda menggunakan transformator input untuk EventBridge mengubah peristiwa sebelum mengirim ke target yang Anda tentukan.

1. Ikuti langkah-langkah dalam ??? prosedur.
2. Pada ??? langkahnya, saat diminta untuk memilih tujuan API sebagai tipe target:
   a. Pilih tujuan EventBridge API.
   b. Lakukan salah satu hal berikut:
      - Pilih Gunakan tujuan API yang ada dan pilih tujuan API yang ada
      - Pilih Buat tujuan API baru dan tentukan setelan yang diperlukan untuk menentukan tujuan API baru Anda.

      Untuk informasi selengkapnya tentang menentukan pengaturan yang diperlukan, lihat???.
   c. Tentukan parameter header Content-Type yang diperlukan untuk acara: CloudEvents
      - Di bawah Parameter Header pilih Tambahkan parameter header.
      - Untuk kunci, tentukan`Content-Type`.

      Untuk nilai, tentukan`application/cloudevents+json; charset=UTF-8`.
3. Tentukan peran eksekusi untuk target Anda.
4. Tentukan transformator input untuk mengubah data peristiwa sumber ke dalam CloudEvents format:
   a. Di bawah Pengaturan tambahan, untuk Konfigurasi input target, pilih Transformator input.

      Kemudian pilih Konfigurasi transformator input.
   b. Di bawah Transformator input target, tentukan jalur Input.

      Di jalur input di bawah ini, atribut region adalah atribut ekstensi kustom dari CloudEvents format. Karena itu tidak diperlukan untuk kepatuhan terhadap CloudEvents spesifikasi.

      CloudEvents memungkinkan Anda untuk menggunakan dan membuat atribut ekstensi yang tidak ditentukan dalam spesifikasi inti. Untuk informasi selengkapnya, termasuk daftar atribut

ekstensi yang diketahui, lihat Atribut CloudEvents Ekstensi dalam dokumentasi CloudEvents spesifikasi pada GitHub.

```
{
  "detail": "$.detail",
  "detail-type": "$.detail-type",
  "id": "$.id",
  "region": "$.region",
  "source": "$.source",
  "time": "$.time"
}
```

c. Untuk Template, masukkan template untuk mengubah data peristiwa sumber ke CloudEvents format.

   Dalam template di bawah `region` ini, tidak sepenuhnya diperlukan, karena `region` atribut di jalur input adalah atribut ekstensi untuk CloudEvents spesifikasi.

```
{
  "specversion":"1.0",
  "id":<id>,
  "source":<source>,
  "type":<detail-type>,
  "time":<time>,
  "region":<region>,
  "data":<detail>
}
```

5. Selesaikan pembuatan aturan mengikuti langkah-langkah prosedur.

Mitra tujuan API

Gunakan informasi yang diberikan oleh AWS Mitra berikut untuk mengonfigurasi tujuan API dan koneksi untuk layanan atau aplikasi mereka.

Konfluen

URL titik akhir pemanggilan tujuan API:

Biasanya format berikut:

```
https://random-id.region.aws.confluent.cloud:443/kafka/v3/
clusters/cluster-id/topics/topic-name/records
```

Untuk informasi selengkapnya, lihat Menemukan alamat endpoint REST dan ID cluster dalam dokumentasi Confluent.

Jenis otorisasi yang didukung:

Basic

Parameter otorisasi tambahan diperlukan:

Tidak berlaku

Dokumentasi pertemuan:

Menghasilkan Catatan

Proxy REST Confluent untuk Apache Kafka

Operasi API yang umum digunakan:

`POST`

Informasi tambahan:

Untuk mengubah data peristiwa menjadi pesan yang dapat diproses oleh titik akhir, buat transformator input target.

- Untuk menghasilkan rekaman tanpa menentukan kunci partisi Kafka, gunakan template berikut untuk transformator input Anda. Tidak diperlukan jalur input.

```
{
  "value":{
      "type":"JSON",
      "data":aws.events.event.json
  },
}
```

- Untuk menghasilkan catatan menggunakan bidang data peristiwa sebagai kunci partisi Kafka, ikuti jalur input dan contoh templat di bawah ini. Contoh ini mendefinisikan jalur input untuk `orderId` bidang, dan kemudian menentukan bidang itu sebagai kunci partisi.

  Pertama, tentukan jalur input untuk bidang data peristiwa:

```
{
```

```
  "orderId":"$.detail.orderId"
}
```

Kemudian, gunakan template transformator input untuk menentukan bidang data sebagai kunci partisi:

```
{
  "value":{
    "type":"JSON",
    "data":aws.events.event.json
  },
  "key":{
    "data":"<orderId>",
    "type":"STRING"
  }
}
```

Coralogix

URL titik akhir pemanggilan tujuan API

Untuk daftar lengkap titik akhir, lihat Referensi Coralogix API.

Jenis otorisasi yang didukung

Kunci API

Parameter otorisasi tambahan diperlukan

Header"`x-amz-event-bridge-access-key`", nilainya adalah Kunci Coralogix API

Dokumentasi Coralogix

EventBridgeAutentikasi Amazon

Operasi API yang umum digunakan

AS: https://ingress.coralogix.us/aws/event-bridge

Singapura: https://ingress.coralogixsg.com/aws/event-bridge

Irlandia: https://ingress.coralogix.com/aws/event-bridge

Oslo: https://ingress.eu2.coralogix.com/aws/event-bridge

Indonesia: https://ingress.coralogix.in/aws/event-bridge

Informasi tambahan

Peristiwa disimpan sebagai entri log dengan `applicationName=[AWS Account]` dan`subsystemName=[event.source]`.

Datadog

URL titik akhir pemanggilan tujuan API

Untuk daftar lengkap titik akhir, lihat Referensi Datadog API.

Jenis otorisasi yang didukung

Kunci API

Parameter otorisasi tambahan diperlukan

Tidak ada

Dokumentasi Datadog

Autentikasi

Operasi API yang umum digunakan

POST https://api.datadoghq.com/api/v1/events

POST https://http-intake.logs.datadoghq.com/v1/input

Informasi tambahan

URL titik akhir berbeda bergantung pada lokasi organisasi Datadog Anda. Untuk URL yang benar untuk organisasi Anda, lihat dokumentasi.

Freshworks

URL titik akhir pemanggilan tujuan API

Untuk daftar titik akhir, lihat https://developers.freshworks.com/documentation/

Jenis otorisasi yang didukung

Basic, Kunci API

Parameter otorisasi tambahan diperlukan

Tidak berlaku

Dokumentasi Freshworks

Autentikasi

Operasi API yang umum digunakan

https://developers.freshdesk.com/api/#create_ticket

https://developers.freshdesk.com/api/#update_ticket

https://developer.freshsales.io/api/#create_lead

https://developer.freshsales.io/api/#update_lead

Informasi tambahan

Tidak ada

MongoDB

URL titik akhir pemanggilan tujuan API

https://data.mongodb-api.com/app/ *ID aplikasi/titik akhir*/

Jenis otorisasi yang didukung

Kunci API

Email/Kata Sandi

Otentikasi JWT Kustom

Parameter otorisasi tambahan diperlukan

Tidak ada

Dokumentasi MongoDB

Atlas Data API

Titik akhir

Titik Akhir HTTPS Kustom

Autentikasi

Operasi API yang umum digunakan

Tidak ada

Informasi tambahan

Tidak ada

New Relic

URL titik akhir pemanggilan tujuan API

Untuk informasi selengkapnya, lihat Pusat data wilayah Uni Eropa dan AS.

Peristiwa

AS– https://insights-collector.newrelic.com/v1/accounts/*YOUR_NEW_RELIC_ACCOUNT_ID*/events

UE– https://insights-collector.eu01.nr-data.net/v1/accounts/*YOUR_NEW_RELIC_ACCOUNT_ID*/events

Metrik-metrik

AS– https://metric-api.newrelic.com/metric/v1

UE– https://metric-api.eu.newrelic.com/metric/v1

Log

AS– https://log-api.newrelic.com/log/v1

UE– https://log-api.eu.newrelic.com/log/v1

Jejak

AS– https://trace-api.newrelic.com/trace/v1

UE– https://trace-api.eu.newrelic.com/trace/v1

Jenis otorisasi yang didukung

- Kunci API

Dokumentasi New Relic

- Metrik API
- API Acara
- Log API
- API Jejak

Operasi API yang umum digunakan

- Metrik API
- API Acara
- Log API
- API Jejak

Informasi tambahan

- Batas API metrik
- Batas API peristiwa
- Batas API log
- Lacak batas API

Operata

URL titik akhir pemanggilan tujuan API:

- https://api.operata.io/v2/aws/events/contact-record

Jenis otorisasi yang didukung:

- Basic

Parameter otorisasi tambahan diperlukan:

Tidak ada

Dokumentasi Operata:

Bagaimana cara membuat, melihat, mengubah, dan mencabut Token API?

AWS Integrasi Operata menggunakan Amazon EventBridge Scheduler Pipes

Operasi API yang umum digunakan:

```
POST https://api.operata.io/v2/aws/events/contact-record
```

Informasi tambahan:

`username`Ini adalah ID Grup Operata dan kata sandi adalah token API Anda.

Salesforce

URL titik akhir pemanggilan tujuan API

Sobject- *`https://.my.salesforce.com/services/data/ versionNumber / sobjects//* myDomainNameSobjectEndpoint`*

Acara platform kustom- *`https://myDomainName.my.salesforce.com/services/data/ versionNumber /sobjects/ /* customPlatformEndpoint`*

Untuk daftar lengkap titik akhir, lihat Referensi SalesforceAPI

Jenis otorisasi yang didukung

Kredenensi klien OAuth

Token OAUTH disegarkan saat respons 401 atau 407 dikembalikan.

Parameter otorisasi tambahan diperlukan

SalesforceId Klien Aplikasi Terhubung dan Rahasia Klien.

Salah satu titik akhir otorisasi berikut:

- Produksi- https://*`MyDomainName`*.my.salesforce.com. /layanan/oauth2/token
- Kotak pasir tanpa domain yang disempurnakan— https://*`MyDomainName`*-- *`SandboxName`*.my.salesforce.com/services /oauth2/token

- Kotak pasir dengan domain yang disempurnakan— https://-- .sandbox.my.salesforce.com/services/oauth2/token ***MyDomainName SandboxName***

Pasangan kunci/nilai berikut:

Kunci	Nilai
grant_type	client_credentials

Dokumentasi Salesforce

Panduan Pengembang REST API

Operasi API yang umum digunakan

Bekerja dengan Object Metadata

Bekerja dengan Rekaman

Informasi tambahan

Untuk tutorial yang menjelaskan cara menggunakan EventBridge konsol untuk membuat koneksi keSalesforce, Tujuan API, dan aturan untuk merutekan informasiSalesforce, lihat???.

Slack

URL titik akhir pemanggilan tujuan API

Untuk daftar titik akhir dan sumber daya lainnya, lihat Menggunakan Slack Web API

Jenis otorisasi yang didukung

OAuth 2.0

Token OAUTH disegarkan saat respons 401 atau 407 dikembalikan.

Saat Anda membuat Slack aplikasi dan menginstalnya ke ruang kerja Anda, token pembawa OAuth akan dibuat atas nama Anda untuk digunakan untuk mengautentikasi panggilan oleh koneksi tujuan API Anda.

Parameter otorisasi tambahan diperlukan

Tidak berlaku

Dokumentasi Slack

Pengaturan aplikasi dasar

Instalasi dengan OAuth

Mengambil pesan

Mengirim pesan

Mengirim pesan menggunakan Webhook Masuk

Operasi API yang umum digunakan

https://slack.com/api/chat.postMessage

Informasi tambahan

Saat mengonfigurasi EventBridge aturan Anda, ada dua konfigurasi untuk disorot:

- Sertakan parameter header yang mendefinisikan jenis konten sebagai “application/json; charset=utf-8".
- Gunakan transformator input untuk memetakan peristiwa input ke output yang diharapkan untuk Slack API, yaitu memastikan bahwa muatan yang dikirim ke Slack API memiliki pasangan kunci/nilai “channel” dan “text”.

Shopify

URL titik akhir pemanggilan tujuan API

Untuk daftar titik akhir dan sumber dan metode lainnya, lihat Titik akhir dan permintaan

Jenis otorisasi yang didukung

OAuth, Kunci API

> **Note**
>
> Token OAUTH disegarkan saat respons 401 atau 407 dikembalikan.

Parameter otorisasi tambahan diperlukan

Tidak berlaku

Dokumentasi Shopify

Ikhtisar otentikasi dan otorisasi

Operasi API yang umum digunakan

POSTING - /admin/api/2022-01/products.json

DAPATKAN - admin/api/2022-01/products/ {product_id} .json

PUT - admin/api/2022-01/products/ {product_id} .json

HAPUS - admin/api/2022-01/products/ {product_id} .json

Informasi tambahan

Buat aplikasi

Pengiriman EventBridge webhook Amazon

Akses token untuk aplikasi kustom di Shopify admin

Produk

ShopifyAdmin API

Splunk

URL titik akhir pemanggilan tujuan API

https://*SPLUNK_HEC_ENDPOINT*:*optional_port*/services/collector/raw

Jenis otorisasi yang didukung

Basic, Kunci API

Parameter otorisasi tambahan diperlukan

Tidak ada

Dokumentasi Splunk

Untuk kedua jenis otorisasi, Anda memerlukan ID token HEC. Untuk informasi selengkapnya, lihat Mengatur dan menggunakan HTTP Event Collector di Splunk Web.

Operasi API yang umum digunakan

POST https://*SPLUNK_HEC_ENDPOINT*:*optional_port*/services/collector/raw

Informasi tambahan

Kunci API — Saat mengonfigurasi titik akhir untuk EventBridge, nama kunci API adalah “Otorisasi” dan nilainya adalah ID token Splunk HEC.

Dasar (Nama Pengguna/Kata Sandi) — Saat mengonfigurasi titik akhir untuk EventBridge, nama pengguna adalah “Splunk” dan kata sandi adalah ID token Splunk HEC.

Sumo Logic

URL titik akhir pemanggilan tujuan API

HTTP Log dan Metric Source endpoint URL akan berbeda untuk setiap pengguna. Untuk informasi selengkapnya, lihat Log HTTP dan Sumber Metrik.

Jenis otorisasi yang didukung

Sumo Logictidak memerlukan otentikasi pada Sumber HTTP mereka karena ada kunci unik yang dimasukkan ke dalam URL. Untuk alasan ini, Anda harus memastikan untuk memperlakukan URL itu sebagai rahasia.

Saat Anda mengonfigurasi tujuan EventBridge API, jenis otorisasi diperlukan. Untuk memenuhi persyaratan ini, pilih API Key dan berikan nama kunci “dummy-key” dan nilai kunci “dummy-value”.

Parameter otorisasi tambahan diperlukan

Tidak berlaku

Dokumentasi Sumo Logic

Sumo Logictelah membangun sumber yang dihosting untuk mengumpulkan log dan metrik dari banyak AWS layanan dan Anda dapat menggunakan informasi di situs web mereka untuk bekerja dengan sumber-sumber tersebut. Untuk informasi selengkapnya, lihat Amazon Web Services.

Jika Anda membuat peristiwa khusus dari aplikasi dan ingin mengirimkannya Sumo Logic sebagai log atau metrik, gunakan Tujuan EventBridge API dan Log Sumo Logic HTTP dan titik akhir Sumber Metrik.

- Untuk mendaftar dan membuat Sumo Logic instans gratis, lihat Mulai uji coba gratis Anda hari ini.
- Untuk informasi selengkapnya tentang penggunaanSumo Logic, lihat Log HTTP dan Sumber Metrik.

Operasi API yang umum digunakan

POSTING https://endpoint4.collection.us2.sumologic.com/receiver/v1/http/*UNIQUE_ID_PER_COLLECTOR*

Informasi tambahan

Tidak ada

TriggerMesh

URL titik akhir pemanggilan tujuan API

Gunakan informasi dalam topik Sumber Peristiwa untuk HTTP untuk merumuskan URL titik akhir. URL titik akhir mencakup nama sumber peristiwa dan namespace pengguna dalam format berikut ini:

https://*source-name*.*user-namespace*.cloud.triggermesh.io

Mencakup parameter otorisasi Basic dalam permintaan ke titik akhir.

Jenis otorisasi yang didukung

Basic

Parameter otorisasi tambahan diperlukan

Tidak ada

Dokumentasi TriggerMesh

Sumber Event untuk HTTP

Operasi API yang umum digunakan

Tidak berlaku

Informasi tambahan

Tidak ada

Zendesk

URL titik akhir pemanggilan tujuan API

https://developer.zendesk.com/rest_api/docs/support/tickets

Jenis otorisasi yang didukung

Basic, Kunci API

Parameter otorisasi tambahan diperlukan

Tidak ada

Dokumentasi Zendesk

Keamanan dan Otentikasi

Operasi API yang umum digunakan

POST https://*your_Zendesk_subdomain*/api/v2/tickets

Informasi tambahan

Permintaan EventBridge API dihitung terhadap batas API Zendesk Anda. Untuk informasi tentang batas Zendesk untuk rencana Anda, lihat Batas penggunaan.

Untuk melindungi akun dan data Anda dengan lebih baik, sebaiknya gunakan kunci API daripada autentikasi kredensi login dasar.

EventBridge Target Amazon untuk Amazon API Gateway

Anda dapat menggunakan Amazon API Gateway untuk membuat, menerbitkan, memelihara, dan memantau API. Amazon EventBridge mendukung pengiriman peristiwa ke titik akhir API Gateway. Saat Anda menentukan titik akhir API Gateway sebagai target, setiap peristiwa dikirim ke peta target untuk permintaan yang dikirim ke titik akhir.

Important

EventBridge mendukung penggunaan API Gateway Edge-optimized dan endpoint Regional sebagai target. Titik akhir pribadi saat ini tidak didukung. Untuk mempelajari selengkapnya tentang titik akhir, lihat https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-api-endpoint-types.html.

Anda dapat menggunakan target API Gateway untuk kasus penggunaan berikut ini:

- Untuk memanggil API yang ditentukan pelanggan yang dihosting di API Gateway berdasarkan peristiwa AWS atau pihak ketiga.
- Untuk memanggil titik akhir secara berkala pada jadwal.

Informasi peristiwa EventBridge JSON dikirim sebagai badan permintaan HTTP ke titik akhir Anda. Anda dapat menentukan atribut permintaan lainnya di bidang `HttpParameters` target sebagai berikut:

- Daftar `PathParameterValues` adalah nilai yang sesuai secara berurutan untuk setiap variabel jalur dalam ARN titik akhir Anda, sebagai contoh `"arn:aws:execute-api:us-east-1:112233445566:myapi/dev/POST/pets/*/*"`.
- `QueryStringParameters`mewakili parameter string kueri yang EventBridge ditambahkan ke titik akhir yang dipanggil.
- `HeaderParameters` menetapkan header HTTP untuk ditambahkan ke permintaan.

> **Note**
>
> Untuk pertimbangan keamanan, kunci header HTTP berikut ini tidak diizinkan:
>
> - Apa pun diawali dengan `X-Amz` atau `X-Amzn`
> - `Authorization`
> - `Connection`
> - `Content-Encoding`
> - `Content-Length`
> - `Host`
> - `Max-Forwards`
> - `TE`
> - `Transfer-Encoding`
> - `Trailer`
> - `Upgrade`
> - `Via`
> - `WWW-Authenticate`

- `X-Forwarded-For`

Parameter Dinamis

Ketika meminta target API Gateway, secara dinamis Anda dapat menambahkan data ke peristiwa yang dikirim ke target. Untuk informasi selengkapnya, lihat the section called “Parameter terget”.

Coba Lagi Invokasi

Seperti semua target, EventBridge coba ulang beberapa pemanggilan yang gagal. Untuk API Gateway, EventBridge coba ulang respons yang dikirim dengan kode status HTTP 5xx atau 429 hingga 24 jam dengan mundur dan jitter eksponensial. Setelah itu, EventBridge terbitkan `FailedInvocations` metrik di Amazon CloudWatch. EventBridge tidak mencoba lagi kesalahan HTTP 4xx lainnya.

Waktu habis

EventBridge aturan Permintaan API Gateway harus memiliki batas waktu eksekusi klien maksimum 5 detik. Jika API Gateway membutuhkan waktu lebih dari 5 detik untuk merespons, EventBridge hentikan permintaan, lalu coba lagi.

EventBridge Permintaan Pipes API Gateway memiliki batas waktu maksimum 29 detik, maksimum API Gateway.

AWS AppSync target untuk Amazon EventBridge

AWS AppSync memungkinkan pengembang untuk menghubungkan aplikasi dan layanan mereka ke data dan peristiwa dengan GraphQL dan Pub/Sub API yang aman, tanpa server, dan berkinerja tinggi. Dengan AWS AppSync, Anda dapat mempublikasikan pembaruan data real-time ke aplikasi Anda dengan mutasi GraphQL. EventBridge mendukung pemanggilan operasi mutasi GraphQL yang valid untuk peristiwa yang cocok. Saat Anda menentukan mutasi AWS AppSync API sebagai target, AWS AppSync proses peristiwa melalui operasi mutasi, yang kemudian dapat memicu langganan yang ditautkan ke mutasi.

> **Note**
>
> EventBridge mendukung API GraphQL AWS AppSync publik. EventBridge saat ini tidak mendukung API AWS AppSync Pribadi.

Anda dapat menggunakan target AWS AppSync GraphQL API untuk kasus penggunaan berikut:

- Untuk mendorong, mengubah, dan menyimpan data peristiwa ke sumber data yang dikonfigurasi.
- Untuk mengirim notifikasi real-time ke klien aplikasi yang terhubung.

> **Note**
>
> AWS AppSync target hanya mendukung pemanggilan AWS AppSync GraphQL API menggunakan `AWS_IAM` jenis otorisasi.

Untuk informasi selengkapnya tentang AWS AppSync GraphQL API, lihat GraphQL dan arsitektur di Panduan AWS AppSync Pengembang.AWS AppSync

Untuk menentukan AWS AppSync target untuk EventBridge aturan menggunakan konsol

1. Buat atau edit aturan.
2. Di bawah Target, tentukan target dengan memilih AWS layanan dan kemudian AWS AppSync.
3. Tentukan operasi mutasi yang akan diuraikan dan dieksekusi, bersama dengan set seleksi.

   - Pilih AWS AppSync API, lalu mutasi GraphQL API yang akan dipanggil.
   - Di bawah Konfigurasi parameter dan set pilihan, pilih untuk membuat set pilihan menggunakan pemetaan nilai kunci atau transformator input.

     Key-value mapping

     Untuk menggunakan pemetaan nilai kunci untuk membuat set pilihan Anda:

     - Tentukan variabel untuk parameter API. Setiap variabel dapat berupa nilai statis atau ekspresi jalur JSON dinamis ke payload acara.
     - Di bawah Set pilihan, pilih variabel yang ingin Anda sertakan dalam respons.

     Input transformer

     Untuk menggunakan transformator input untuk membuat set pilihan Anda:

     - Tentukan jalur input yang mendefinisikan variabel yang akan digunakan.
     - Tentukan template input untuk menentukan dan memformat informasi yang ingin Anda lewatkan ke target.

     Untuk informasi selengkapnya, lihat ???.

4. Untuk peran Eksekusi, pilih apakah akan membuat peran baru atau menggunakan peran yang sudah ada.
5. Selesaikan membuat atau mengedit aturan.

Contoh: AWS AppSync target untuk Amazon EventBridge

Dalam contoh berikut, kita akan membahas cara menentukan AWS AppSync target untuk EventBridge aturan, termasuk mendefinisikan transformasi input untuk memformat peristiwa untuk pengiriman.

Misalkan Anda memiliki AWS AppSync GraphQL API`Ec2EventAPI`,, yang ditentukan oleh skema berikut:

```
type Event {
    id: ID!
    statusCode: String
    instanceId: String
}

type Mutation {
    pushEvent(id: ID!, statusCode: String!, instanceId: String): Event
}

type Query {
    listEvents: [Event]
}

type Subscription {
    subscribeToEvent(id: ID, statusCode: String, instanceId: String): Event
        @aws_subscribe(mutations: ["pushEvent"])
}
```

Klien aplikasi yang menggunakan API ini dapat `subscribeToEvent` berlangganan langganan, yang dipicu oleh `pushEvent` mutasi.

Anda dapat membuat EventBridge aturan dengan target yang mengirimkan peristiwa ke AppSync API melalui `pushEvent` mutasi. Ketika mutasi dipanggil, setiap klien yang berlangganan akan menerima acara tersebut.

Untuk menentukan API ini sebagai target EventBridge aturan, Anda akan melakukan hal berikut:

1. Tetapkan Nama Sumber Daya Amazon (ARN) dari target aturan ke ARN titik akhir GraphQL API. `Ec2EventAPI`
2. Tentukan Operasi GraphQL mutasi sebagai parameter target:

```
mutation CreatePushEvent($id: ID!, $statusCode: String, $instanceId: String) {
  pushEvent(id: $input, statusCode: $statusCode, instanceId: $instanceId) {
    id
    statusCode
    instanceId
  }
}
```

   Set pilihan mutasi Anda harus menyertakan semua bidang yang ingin Anda berlangganan dalam langganan GraphQL Anda.
3. Konfigurasikan transformator input untuk menentukan bagaimana data dari peristiwa yang cocok digunakan dalam operasi Anda.

   Misalkan Anda memilih "`EC2 Instance Launch Successful`" contoh acara:

```
{
  "version": "0",
  "id": "3e3c153a-8339-4e30-8c35-687ebef853fe",
  "detail-type": "EC2 Instance Launch Successful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2015-11-11T21:31:47Z",
  "region": "us-east-1",
  "resources": ["arn:aws:autoscaling:us-
east-1:123456789012:autoScalingGroup:eb56d16b-bbf0-401d-b893-
d5978ed4a025:autoScalingGroupName/sampleLuanchSucASG", "arn:aws:ec2:us-
east-1:123456789012:instance/i-b188560f"],
  "detail": {
    "StatusCode": "InProgress",
    "AutoScalingGroupName": "sampleLuanchSucASG",
    "ActivityId": "9cabb81f-42de-417d-8aa7-ce16bf026590",
    "Details": {
      "Availability Zone": "us-east-1b",
      "Subnet ID": "subnet-95bfcebe"
    },
    "RequestId": "9cabb81f-42de-417d-8aa7-ce16bf026590",
    "EndTime": "2015-11-11T21:31:47.208Z",
```

```
    "EC2InstanceId": "i-b188560f",
    "StartTime": "2015-11-11T21:31:13.671Z",
    "Cause": "At 2015-11-11T21:31:10Z a user request created an AutoScalingGroup
 changing the desired capacity from 0 to 1.  At 2015-11-11T21:31:11Z an instance was
 started in response to a difference between desired and actual capacity, increasing
 the capacity from 0 to 1."
  }
}
```

Anda dapat menentukan variabel berikut untuk digunakan dalam template Anda, menggunakan jalur input transformator input target:

```
{
  "id": "$.id",
  "statusCode": "$.detail.StatusCode",
  "EC2InstanceId": "$.detail.EC2InstanceId"
}
```

Tulis template transformator input untuk menentukan variabel yang EventBridge lolos ke operasi AWS AppSync mutasi. Template harus mengevaluasi ke JSON. Dengan jalur masukan kami, Anda dapat membuat template berikut:

```
{
  "id": <id>,
  "statusCode": <statusCode>,
  "instanceId": <EC2InstanceId>
}
```

Koneksi untuk target titik akhir HTTP

Koneksi mendefinisikan metode otorisasi dan kredensyal untuk digunakan dalam menghubungkan EventBridge ke titik akhir HTTP tertentu. Ketika Anda mengkonfigurasi pengaturan otorisasi dan membuat koneksi, itu menciptakan rahasia AWS Secrets Manager untuk menyimpan informasi otorisasi dengan aman. Anda juga dapat menambahkan parameter tambahan untuk disertakan dalam koneksi yang sesuai untuk target titik akhir HTTP Anda.

Gunakan koneksi dengan:

- Tujuan API

Saat Anda membuat tujuan API, Anda menentukan koneksi yang akan digunakan untuk itu. Anda dapat memilih koneksi yang ada dari akun Anda, atau membuat koneksi saat membuat tujuan API.

Metode otorisasi untuk koneksi

EventBridge koneksi mendukung metode otorisasi berikut:

- Basic
- Kunci API

  Untuk otorisasi Kunci Dasar dan API, EventBridge isi header otorisasi yang diperlukan untuk Anda.
- OAuth

  Untuk otorisasi OAuth, EventBridge juga menukar ID klien dan rahasia Anda dengan token akses dan kemudian mengelolanya dengan aman.

  Token OAUTH disegarkan saat respons 401 atau 407 dikembalikan.

Ketika Anda membuat koneksi, Anda juga dapat menyertakan header, bodi, dan parameter kueri yang diperlukan untuk otorisasi dengan titik akhir. Anda dapat menggunakan koneksi yang sama untuk lebih dari satu titik akhir HTTP jika otorisasi untuk titik akhir sama.

Saat Anda membuat koneksi dan menambahkan parameter otorisasi, EventBridge buat rahasia di AWS Secrets Manager. Biaya penyimpanan dan akses rahasia Secrets Manager disertakan dengan biaya untuk menggunakan tujuan API. Untuk mempelajari lebih lanjut tentang praktik terbaik untuk menggunakan rahasia dengan tujuan API, lihat AWS::Events::ApiDestinationdi Panduan CloudFormation Pengguna.

> **Note**
>
> Agar berhasil membuat atau memperbarui koneksi, Anda harus menggunakan akun yang memiliki izin untuk menggunakan Secrets Manager. Izin yang diperlukan disertakan dalam AmazonEventBridgeFullAccess kebijakan. Izin yang sama diberikan kepada peran tertaut layanan yang dibuat di akun Anda untuk koneksi.

Membuat koneksi untuk target titik akhir HTTP

Untuk membuat koneksi untuk digunakan dengan titik akhir HTTP menggunakan konsol EventBridge

1. Masuk untuk AWS menggunakan akun yang memiliki izin untuk mengelola EventBridge dan membuka EventBridge konsol.
2. Di panel navigasi kiri, pilih tujuan API.
3. Gulir ke bawah ke tabel tujuan API, dan kemudian pilih tab Koneksi.
4. Pilih Buat koneksi.
5. Pada halaman Buat koneksi, masukkan Nama koneksi untuk koneksi.
6. Masukkan Deskripsi untuk koneksi.
7. Untuk Jenis otorisasi, pilih jenis otorisasi yang akan digunakan untuk mengotorisasi koneksi ke titik akhir HTTP yang ditentukan untuk tujuan API yang menggunakan koneksi ini. Lakukan salah satu dari berikut ini:
   - Pilih Basic (Nama Pengguna/Kata Sandi), dan kemudian masukkan Nama pengguna dan Kata Sandi untuk digunakan untuk mengotorisasi dengan titik akhir HTTP.
   - Pilih Kredensi Klien OAuth, dan kemudian masukkan Titik akhir Autentikasi, metode HTTP, ID klien, dan Rahasia klien yang akan digunakan untuk mengotorisasi dengan titik akhir.

     Di bawah Parameter Http OAuth, tambahkan parameter tambahan apa pun yang akan disertakan untuk otorisasi dengan titik akhir otorisasi. Pilih Parameter dari daftar menurun, kemudian masukkan Kunci dan Nilai. Untuk memasukkan parameter tambahan, pilih Tambahkan parameter.

     Di bawah Parameter Http invokasi, tambahkan parameter tambahan apa pun yang akan disertakan dalam permintaan otorisasi. Untuk menambahkan parameter, pilih Parameter dari daftar menurun, kemudian masukkan Kunci dan Nilai. Untuk memasukkan parameter tambahan, pilih Tambahkan parameter.
   - Pilih Kunci API, dan kemudian masukkan Nama kunci API dan Nilai yang berkaitan yang akan digunakan untuk otorisasi Kunci API.

     Di bawah Parameter Http invokasi, tambahkan parameter tambahan apa pun yang akan disertakan dalam permintaan otorisasi. Untuk menambahkan parameter, pilih Parameter dari daftar menurun, kemudian masukkan Kunci dan Nilai. Untuk memasukkan parameter tambahan, pilih Tambahkan parameter.
8. Pilih Buat.

Mengedit koneksi menggunakan EventBridge konsol

Anda dapat mengedit koneksi yang ada.

Untuk mengedit koneksi menggunakan EventBridge konsol

1. Masuk untuk AWS menggunakan akun yang memiliki izin untuk mengelola EventBridge dan membuka EventBridge konsol.
2. Di panel navigasi kiri, pilih tujuan API.
3. Gulir ke bawah ke tabel tujuan API, dan kemudian pilih tab Koneksi.
4. Dalam tabel Koneksi, pilih koneksi yang akan diedit.
5. Pada halaman Detail koneksi, pilih Edit.
6. Perbarui nilai untuk koneksi, dan kemudian pilih Perbarui.

De-otorisasi koneksi menggunakan konsol EventBridge

Saat Anda mendeotorisasi koneksi, maka akan menghapus semua parameter otorisasi. Menghapus parameter otorisasi akan menghapus rahasia dari koneksi, sehingga Anda dapat menggunakannya kembali tanpa harus membuat koneksi baru.

> **Note**
>
> Anda harus memperbarui titik akhir HTTP apa pun yang menggunakan koneksi yang tidak diotorisasi untuk menggunakan koneksi yang berbeda agar berhasil mengirim permintaan ke titik akhir HTTP.

Untuk mendeotorisasi koneksi

1. Masuk untuk AWS menggunakan akun yang memiliki izin untuk mengelola EventBridge dan membuka EventBridge konsol.
2. Di panel navigasi kiri, pilih tujuan API.
3. Gulir ke bawah ke tabel tujuan API, dan kemudian pilih tab Koneksi.
4. Dalam tabel Koneksi, pilih koneksi.
5. Pada halaman Detail koneksi, pilih De-otorisasi.

6. Dalam kotak dialog Deotorisasi koneksi?, masukkan nama koneksi, dan kemudian pilih De-otorisasi.

Status koneksi berubah menjadi De-otorisasi hingga proses selesai. Kemudian status berubah menjadi De-otorisasi. Sekarang Anda dapat mengedit koneksi untuk menambahkan parameter otorisasi baru.

Mengirim dan menerima EventBridge acara Amazon antar AWS akun

Anda dapat mengonfigurasi EventBridge untuk mengirim dan menerima acara antara bus acara di AWS akun. Saat Anda mengonfigurasi EventBridge untuk mengirim atau menerima acara antar akun, Anda dapat menentukan AWS akun mana yang dapat mengirim acara atau menerima acara dari bus acara di akun Anda. Anda juga dapat mengizinkan atau menolak acara dari aturan khusus yang terkait dengan bus acara, atau acara dari sumber tertentu. Untuk informasi selengkapnya, lihat Menyederhanakan akses lintas akun dengan kebijakan sumber daya Amazon EventBridge

> **Note**
>
> Jika Anda menggunakannya AWS Organizations, Anda dapat menentukan organisasi dan memberikan akses ke semua akun di organisasi tersebut. Selain itu, bus acara pengirim harus memiliki peran IAM yang melekat padanya saat mengirim acara ke akun lain. Untuk informasi selengkapnya, lihat Apa itu AWS Organizations dalam AWS Organizations Panduan Pengguna.

> **Note**
>
> Jika Anda menggunakan rencana respons Incident Manager sebagai target, semua rencana respons yang dibagikan dengan akun Anda tersedia secara default.

Anda dapat mengirim dan menerima acara antar bus acara di AWS akun dalam Wilayah yang sama di semua Wilayah dan antar akun di Wilayah yang berbeda selama Wilayah tujuan adalah Wilayah tujuan Lintas Wilayah yang didukung.

Langkah-langkah untuk mengonfigurasi EventBridge untuk mengirim acara ke atau menerima acara dari bus acara di akun yang berbeda meliputi:

- Pada akun penerima, edit izin pada bus acara untuk mengizinkan AWS akun tertentu, organisasi, atau semua AWS akun untuk mengirim acara ke akun penerima.
- Pada akun pengirim, atur satu atau lebih aturan yang memiliki bus peristiwa milik akun penerima sebagai target.

  Jika akun pengirim mewarisi izin untuk mengirim peristiwa dari AWS Organisasi, akun pengirim juga harus memiliki peran IAM dengan kebijakan yang memungkinkannya mengirim peristiwa ke akun penerima. Jika Anda menggunakan AWS Management Console untuk membuat aturan yang menargetkan bus acara di akun penerima, peran akan dibuat secara otomatis. Jika Anda menggunakan AWS CLI, Anda harus membuat peran secara manual.
- Pada akun penerima, atur satu atau lebih aturan yang cocok dengan peristiwa yang berasal dari akun pengirim.

Peristiwa yang dikirim dari satu akun ke akun lainnya dibebankan ke akun pengiriman sebagai peristiwa kustom. Akun penerima tidak dikenakan biaya. Untuk informasi selengkapnya, lihat EventBridge Harga Amazon.

Jika akun penerima menetapkan aturan yang mengirimkan peristiwa yang diterima dari akun pengirim ke akun ketiga, peristiwa tersebut tidak dikirim ke akun ketiga.

Video berikut mencakup peristiwa perutean antar akun: Merutekan acara ke bus di akun lain AWS

Berikan izin untuk mengizinkan acara dari akun lain AWS

Untuk menerima acara dari akun atau organisasi lain, Anda harus terlebih dahulu mengedit izin di bus acara tempat Anda ingin menerima acara. Bus acara default menerima acara dari AWS layanan, AWS akun resmi lainnya, dan `PutEvents` panggilan. Izin untuk bus peristiwa diberikan atau ditolak menggunakan kebijakan berbasis sumber daya yang dilampirkan pada bus peristiwa. Dalam kebijakan, Anda dapat memberikan izin ke AWS akun lain menggunakan ID akun, atau ke AWS organisasi yang menggunakan ID organisasi. Untuk mempelajari lebih lanjut tentang izin bus peristiwa, termasuk kebijakan contoh, lihat Izin untuk busEventBridge peristiwa Amazon.

> **Note**
>
> EventBridge sekarang membutuhkan semua target bus acara lintas akun baru untuk menambahkan peran IAM. Ini hanya berlaku untuk target bus acara yang dibuat setelah 2 Maret 2023. Aplikasi yang dibuat tanpa peran IAM sebelum tanggal tersebut tidak

terpengaruh. Namun, sebaiknya tambahkan peran IAM untuk memberi pengguna akses ke sumber daya di akun lain, karena ini memastikan batasan organisasi yang menggunakan Kebijakan Kontrol Layanan (SCP) diterapkan untuk menentukan siapa yang dapat mengirim dan menerima peristiwa dari akun di organisasi Anda.

Important

Jika Anda memilih untuk menerima acara dari semua AWS akun, berhati-hatilah untuk membuat aturan yang hanya cocok dengan acara yang akan diterima dari orang lain. Untuk membuat aturan yang lebih aman, pastikan bahwa pola peristiwa untuk setiap aturan berisi bidang `Account` dengan ID akun dari satu atau lebih akun yang darinya untuk menerima peristiwa. Aturan yang memiliki pola peristiwa yang berisi bidang Akun tidak cocok dengan peristiwa yang dikirim dari akun yang tidak tercantum dalam bidang `Account`. Untuk informasi selengkapnya, lihat EventBridge Acara Amazon.

Aturan untuk acara antar AWS akun

Jika akun Anda diatur untuk menerima acara dari bus acara di AWS akun lain, Anda dapat menulis aturan yang cocok dengan acara tersebut. Tetapkan pola acara aturan agar sesuai dengan acara yang Anda terima dari bus acara di akun lain.

Kecuali Anda menentukan `account` dalam pola peristiwa suatu aturan, aturan akun Anda, baik yang baru maupun yang sudah ada, yang cocok dengan peristiwa yang Anda terima dari bus acara di akun lain akan dipicu berdasarkan peristiwa tersebut. Jika Anda menerima acara dari bus acara di akun lain, dan Anda ingin aturan dipicu hanya pada pola peristiwa itu ketika dibuat dari akun Anda sendiri, Anda harus menambahkan `account` dan menentukan ID akun Anda sendiri ke pola peristiwa aturan.

Jika Anda mengatur AWS akun Anda untuk menerima acara dari bus acara di semua AWS akun, kami sangat menyarankan agar Anda menambahkan `account` setiap EventBridge aturan di akun Anda. Ini mencegah aturan di akun Anda memicu peristiwa dari AWS akun yang tidak dikenal. Saat Anda menentukan bidang `account` dalam aturan, Anda dapat menentukan ID akun dari lebih dari satu akun AWS di bidang.

Untuk memiliki pemicu aturan pada acara yang cocok dari bus acara apa pun di AWS akun yang telah Anda berikan izin, jangan tentukan * di `account` bidang aturan. Melakukannya tidak akan

cocok dengan peristiwa apa pun, karena * tidak pernah muncul di bidang `account` peristiwa. Alih-alih, hanya menghilangkan bidang `account` dari aturan.

Membuat aturan yang mengirim acara antar AWS akun

Menentukan bus acara di akun lain sebagai target adalah bagian dari pembuatan aturan.

Untuk membuat aturan yang mengirim acara ke AWS akun lain menggunakan konsol

1. Ikuti langkah-langkah dalam ??? prosedur.
2. Pada ??? langkah, ketika diminta untuk memilih jenis target:
   a. Pilih bus EventBridge acara.
   b. Pilih bus Acara di akun atau Wilayah yang berbeda.
   c. Untuk bus Event sebagai target, masukkan ARN bus acara yang ingin Anda gunakan.
3. Selesaikan pembuatan aturan mengikuti langkah-langkah prosedur.

Mengirim dan menerima EventBridge acara Amazon antar AWS Wilayah

Anda dapat mengonfigurasi EventBridge untuk mengirim dan menerima acara antar AWS Wilayah. Anda juga dapat mengizinkan atau menolak peristiwa dari Wilayah spesifik, aturan spesifik yang berkaitan dengan bus peristiwa, atau peristiwa dari sumber spesifik. Untuk informasi selengkapnya, lihat Memperkenalkan perutean acara lintas wilayah dengan Amazon EventBridge

Wilayah berikut adalah daerah tujuan yang didukung:

- Wilayah Afrika (Cape Town)
- Wilayah Asia Pacific (Hong Kong)
- Wilayah Asia Pacific (Tokyo)
- Wilayah Asia Pasifik (Seoul)
- Wilayah Asia Pasifik (Osaka)
- Wilayah Asia Pasifik (Mumbai)
- Wilayah Asia Pasifik (Singapura)
- Wilayah Asia Pacific (Sydney)
- Wilayah Kanada (Pusat)
- Wilayah Eropa (Frankfurt)

- Wilayah Eropa (Stockholm)
- Wilayah Eropa (Milan)
- Wilayah Eropa (Irlandia)
- Wilayah Eropa (London)
- Wilayah Eropa (Paris)
- Wilayah Timur Tengah (UEA)
- Wilayah Middle East (Bahrain)
- Wilayah Amerika Selatan (Sao Paulo)
- Wilayah AS Timur (N. Virginia)
- Wilayah US East (Ohio)
- Wilayah US West (N California)
- Wilayah US West (Oregon)
- Wilayah Asia Pasifik (Jakarta)
- Wilayah Asia Pasifik (Melbourne)
- Wilayah Israel (Tel Aviv)

Video berikut mencakup perutean peristiwa antar Wilayah menggunakan https://console.aws.amazon.com/events/, AWS CloudFormation, dan AWS Serverless Application Model: Perutean acara Lintas Wilayah

Membuat aturan yang mengirim acara ke AWS Wilayah yang berbeda

Menentukan bus acara di AWS Wilayah lain sebagai target adalah bagian dari pembuatan aturan.

Untuk membuat aturan yang mengirim acara ke AWS akun lain menggunakan konsol

1. Ikuti langkah-langkah dalam ??? prosedur.
2. Pada ??? langkah, ketika diminta untuk memilih jenis target:
    a. Pilih bus EventBridge acara.
    b. Pilih bus Acara di akun atau Wilayah yang berbeda.
    c. Untuk bus Event sebagai target, masukkan ARN bus acara yang ingin Anda gunakan.

3. Selesaikan pembuatan aturan mengikuti langkah-langkah prosedur.

Mengirim dan menerima EventBridge acara Amazon antara bus acara di akun dan Wilayah yang sama

Anda dapat mengonfigurasi EventBridge untuk mengirim dan menerima acara antara bus acara di AWS akun dan Wilayah yang sama.

Saat Anda mengonfigurasi EventBridge untuk mengirim atau menerima acara antar bus acara, Anda menggunakan peran IAM di bus acara pengirim untuk memberikan izin bus acara pengirim untuk mengirim acara ke bus acara penerima. Anda menggunakan kebijakan Berbasis sumber daya pada bus peristiwa penerima untuk memberikan bus peristiwa penerima izin untuk menerima penerima dari bus peristiwa pengirim. Anda juga dapat mengizinkan atau menolak peristiwa dari bus peristiwa tertentu, aturan spesifik yang berkaitan dengan bus peristiwa, atau peristiwa dari sumber spesifik. Untuk informasi selengkapnya tentang izin bus peristiwa, termasuk kebijakan contoh, lihat Izin untuk busEventBridge peristiwa Amazon.

Langkah-langkah untuk mengonfigurasi EventBridge untuk mengirim acara ke atau menerima acara antar bus acara di akun Anda meliputi:

- Untuk menggunakan IAM role yang ada, Anda perlu memberikan izin bus peristiwa pengirim ke bus peristiwa penerima atau izin bus peristiwa penerima ke bus peristiwa pengirim.
- Pada bus acara pengirim, atur satu atau lebih aturan yang memiliki bus acara penerima sebagai target dan buat peran IAM. Untuk contoh kebijakan yang harus dilampirkan pada peran, lihat???.
- Pada bus peristiwa penerima, edit izin untuk memungkinkan peristiwa yang akan diteruskan dari bus peristiwa lainnya.
- Pada peristiwa penerima, atur satu atau lebih aturan yang cocok dengan peristiwa yang berasal dari bus peristiwa pengirim.

> **Note**
>
> EventBridge tidak dapat merutekan acara yang diterima dari bus acara pengirim ke bus acara ketiga.

Peristiwa yang dikirim dari satu bus peristiwa ke yang lainnya dibebankan sebagai peristiwa kustom. Untuk informasi lebih lanjut, lihat Amazon EventBridge Harga.

Membuat aturan yang mengirim acara ke bus acara yang berbeda di AWS akun dan Wilayah yang sama

Untuk mengirim peristiwa ke bus peristiwa lainnya, Anda membuat aturan dengan bus peristiwa sebagai target. Menentukan bus acara di AWS akun dan Wilayah yang sama sebagai target adalah bagian dari pembuatan aturan.

Untuk membuat aturan yang mengirim acara ke bus acara yang berbeda di AWS akun dan Wilayah yang sama menggunakan konsol

1. Ikuti langkah-langkah dalam ??? prosedur.
2. Pada ??? langkah, ketika diminta untuk memilih jenis target:
    a. Pilih bus EventBridge acara.
    b. Pilih Bus acara di AWS akun dan Wilayah yang sama.
    c. Untuk bus acara sebagai target, pilih bus acara dari daftar drop-down.
3. Selesaikan pembuatan aturan mengikuti langkah-langkah prosedur.

Transformasi EventBridge masukan Amazon

Anda dapat menyesuaikan teks dari suatu peristiwa sebelum EventBridge meneruskan informasi ke target aturan. Dengan menggunakan pengubah masukan di konsol atau API, Anda menentukan variabel yang menggunakan jalur JSON untuk nilai referensi dalam sumber peristiwa asli. Acara yang diubah dikirim ke target, bukan acara asli. Namun, parameter jalur dinamis harus mereferensikan peristiwa asli, bukan peristiwa yang diubah. Anda dapat menentukan hingga 100 variabel, menetapkan setiap nilai dari masukan. Kemudian Anda dapat menggunakan variabel tersebut di Templat Masukan sebagai <*nama-variabel*>.

Untuk tutorial tentang menggunakan pengubah masukan, lihat ???.

> **Note**
>
> EventBridge tidak mendukung semua sintaks JSON Path dan mengevaluasinya saat runtime. Sintaks yang didukung meliputi:
>
> - notasi titik (misalnya,`$.detail`)
> - tanda hubung
> - menggarisbawahi
> - karakter alfanumerik
> - indeks array
> - wildcard (*)

Dalam topik ini:

- Variabel yang telah ditetapkan
- Contoh transformasi masukan
- Mengubah input dengan menggunakan API EventBridge
- Mengubah masukan dengan menggunakan AWS CloudFormation
- Masalah Umum dengan mengubah input
- Mengkonfigurasi transformator input sebagai bagian dari pembuatan aturan
- Menguji transformator input target menggunakan EventBridge Sandbox

Variabel yang telah ditetapkan

Terdapat variabel yang telah ditetapkan dapat Anda gunakan tanpa menentukan jalur JSON. Variabel tersebut dicadangkan, dan Anda tidak dapat membuat variabel dengan nama tersebut:

- `aws.events.rule-arn`— Nama Sumber Daya Amazon (ARN) dari aturan tersebut. EventBridge
- `aws.events.rule-name`— Nama EventBridge aturan.
- `aws.events.event.ingestion-time`— Waktu di mana acara diterima oleh EventBridge. Ini adalah stempel waktu ISO 8601. Variabel ini dihasilkan oleh EventBridge dan tidak dapat ditimpa.
- `aws.events.event`— Payload acara asli sebagai JSON (tanpa `detail` bidang). Hanya dapat digunakan sebagai nilai untuk bidang JSON, karena isinya tidak lolos.
- `aws.events.event.json`— Muatan acara asli lengkap sebagai JSON. (dengan `detail` lapangan). Hanya dapat digunakan sebagai nilai untuk bidang JSON, karena isinya tidak lolos.

Contoh transformasi masukan

Berikut ini adalah peristiwa Amazon EC2 contoh.

```
{
  "version": "0",
  "id": "7bf73129-1428-4cd3-a780-95db273d1602",
  "detail-type": "EC2 Instance State-change Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2015-11-11T21:29:54Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2:us-east-1:123456789012:instance/i-abcd1111"
  ],
  "detail": {
    "instance-id": "i-0123456789",
    "state": "RUNNING"
  }
}
```

Ketika menetapkan aturan di konsol, pilih opsi Pengubah Masukan di bawah Konfigurasi masukan. Opsi ini menampilkan dua kotak teks: satu untuk Jalur Masukan dan satu untuk Templat Masukan.

Jalur Masukan digunakan untuk menetapkan variabel. Gunakan jalur JSON untuk item referensi dalam peristiwa Anda dan menyimpan nilai tersebut dalam variabel. Sebagai contoh, Anda dapat membuat Jalur Masukan untuk nilai referensi dalam peristiwa contoh dengan memasukkan hal berikut dalam kotak teks pertama. Anda juga dapat menggunakan tanda kurung dan indeks untuk mendapatkan item dari array.

> **Note**
>
> EventBridge menggantikan transformator input saat runtime untuk memastikan output JSON yang valid. Karena itu, letakkan tanda kutip di sekitar variabel yang merujuk ke parameter jalur JSON, tetapi jangan beri tanda kutip di sekitar variabel yang merujuk ke objek atau array JSON.

```
{
  "timestamp" : "$.time",
  "instance" : "$.detail.instance-id",
  "state" : "$.detail.state",
  "resource" : "$.resources[0]"
}
```

Ini mendefinisikan empat variabel,,`<timestamp>`, `<instance><state>`, dan`<resource>`. Anda dapat mereferensikan variabel tersebut saat Anda membuat Templat Masukan.

Templat Masukan adalah templat untuk informasi yang ingin Anda lewatkan ke target Anda. Anda dapat membuat templat yang melewati baik string maupun JSON ke target. Dengan menggunakan peristiwa sebelumnya dan Jalur Masukan, contoh Templat Masukan berikut ini akan mengubah peristiwa untuk keluaran contoh sebelum merutekan ke target.

Deskripsi	Templat	Output
String sederhana	`"instance <instance> is in <state>"`	`"instance i-0123456789 is in RUNNING"`
String dengan tanda kutip lolos	`"instance \"<instance> \" is in <state>"`	`"instance \"i-01234 56789\" is in RUNNING"`

Deskripsi	Templat	Output
		Perhatikan bahwa ini adalah perilaku di EventBridge konsol. AWS CLI keluar dari karakter garis miring dan hasilnya adalah `"instance "i-0123456789" is in RUNNING"`.
JSON sederhana	`{ "instance" : <instance>, "state": <state> }`	`{ "instance" : "i-0123456789", "state": "RUNNING" }`
JSON dengan string dan variabel	`{ "instance" : <instance>, "state": "<state>", "instanceStatus": "instance \"<instance>\" is in <state>" }`	`{ "instance" : "i-0123456789", "state": "RUNNING", "instanceStatus": "instance \"i-0123456789\" is in RUNNING" }`
JSON dengan campuran variabel dan informasi statis	`{ "instance" : <instance>, "state": [ 9, <state>, true ], "Transformed" : "Yes" }`	`{ "instance" : "i-0123456789", "state": [ 9, "RUNNING", true ], "Transformed" : "Yes" }`

Deskripsi	Templat	Output
Termasuk variabel cadangan di JSON	`{ "instance" : <instance>, "state": <state>, "ruleArn" : <aws.events.rule-arn>, "ruleName" : <aws.events.rule-name>, "originalEvent" : <aws.events.event.json> }`	`{ "instance" : "i-0123456789", "state": "RUNNING", "ruleArn" : "arn:aws:events:us-east-2:123456789012:rule/example", "ruleName" : "example", "originalEvent" : { ... // commented for brevity } }`
Termasuk variabel cadangan dalam string	`"<aws.events.rule-name> triggered"`	`"example triggered"`
Grup CloudWatch log Amazon	`{ "timestamp" : <timestamp>, "message": "instance \"<instance>\" is in <state>" }`	`{ "timestamp" : 2015-11-11T21:29:54Z, "message": "instance "i-0123456789" is in RUNNING }`

Mengubah input dengan menggunakan API EventBridge

Untuk informasi tentang penggunaan EventBridge API untuk mengubah input, lihat Menggunakan Input Transformer untuk mengekstrak data dari suatu peristiwa dan memasukkan data tersebut ke target.

Mengubah masukan dengan menggunakan AWS CloudFormation

Untuk informasi tentang menggunakan AWS CloudFormation untuk mengubah input, lihat AWS::Events::Rule InputTransformer.

Masalah Umum dengan mengubah input

Ini adalah beberapa masalah umum saat mengubah input di EventBridge:

- Untuk String, tanda kutip diperlukan.
- Tidak ada validasi saat membuat jalur JSON untuk templat Anda.
- Jika Anda menentukan variabel untuk mencocokkan jalur JSON yang tidak ada dalam peristiwa, variabel tersebut tidak dibuat dan tidak akan muncul dalam keluaran.
- Properti JSON seperti hanya `aws.events.event.json` dapat digunakan sebagai nilai bidang JSON, tidak sebaris di string lain.
- EventBridge tidak lolos dari nilai yang diekstraksi oleh Jalur Input, saat mengisi Template Input untuk target.
- Jika jalur JSON mereferensikan objek atau array JSON, tetapi variabel direferensikan dalam string, EventBridge menghapus tanda kutip internal untuk memastikan string yang valid. Misalnya, untuk variabel yang `<detail>` ditunjuk`$.detail`, “Detail is<detail>” akan mengakibatkan EventBridge penghapusan tanda kutip dari objek.

  Oleh karena itu, jika Anda ingin menampilkan objek JSON berdasarkan variabel jalur JSON tunggal, Anda harus menempatkannya sebagai kunci. Dalam contoh ini,`{"detail": <detail>}`.
- Kutipan tidak diperlukan untuk variabel yang mewakili string. Mereka diizinkan, tetapi EventBridge secara otomatis menambahkan tanda kutip ke nilai variabel string selama transformasi, untuk memastikan output transformasi adalah JSON yang valid. EventBridge tidak menambahkan tanda kutip ke variabel yang mewakili objek atau array JSON. Jangan menambahkan tanda kutip untuk variabel yang mewakili objek JSON atau array.

  Misalnya, template input berikut mencakup variabel yang mewakili string dan objek JSON:

```
{
  "ruleArn" : <aws.events.rule-arn>,
  "ruleName" : <aws.events.rule-name>,
  "originalEvent" : <aws.events.event.json>
```

```
}
```

Menghasilkan JSON yang valid dengan kutipan yang tepat:

```
{
  "ruleArn" : "arn:aws:events:us-east-2:123456789012:rule/example",
  "ruleName" : "example",
  "originalEvent" : {
    ... // commented for brevity
  }
}
```

- Untuk output teks (non-JSON) sebagai string multi-baris, bungkus setiap baris terpisah di template input Anda dalam tanda kutip ganda.

  Misalnya, jika Anda mencocokkan Amazon Inspector Finding event dengan pola event berikut:

```
{
  "detail": {
    "severity": ["HIGH"],
    "status": ["ACTIVE"]
  },
  "detail-type": ["Inspector2 Finding"],
  "source": ["inspector2"]
}
```

  Dan menggunakan jalur input berikut:

```
{
  "account": "$.detail.awsAccountId",
  "ami": "$.detail.resources[0].details.awsEc2Instance.imageId",
  "arn": "$.detail.findingArn",
  "description": "$.detail.description",
  "instance": "$.detail.resources[0].id",
  "platform": "$.detail.resources[0].details.awsEc2Instance.platform",
  "region": "$.detail.resources[0].region",
  "severity": "$.detail.severity",
  "time": "$.time",
  "title": "$.detail.title",
  "type": "$.detail.type"
}
```

Anda dapat menggunakan template input di bawah ini untuk menghasilkan output string multi-baris:

```
"<severity> severity finding <title>"
"Description: <description>"
"ARN: \"<arn>\""
"Type: <type>"
"AWS Account: <account>"
"Region: <region>"
"EC2 Instance: <instance>"
"Platform: <platform>"
"AMI: <ami>"
```

Mengkonfigurasi transformator input sebagai bagian dari pembuatan aturan

Sebagai bagian dari pembuatan aturan, Anda dapat menentukan transformator input EventBridge untuk digunakan untuk memproses peristiwa pencocokan sebelum mengirim peristiwa tersebut ke target yang ditentukan. Anda dapat mengonfigurasi transformator input untuk target yang merupakan AWS layanan atau tujuan API.

Untuk membuat transformator input target sebagai bagian dari aturan

1. Ikuti langkah-langkah untuk membuat aturan seperti yang dijelaskan dalam???.
2. Di Langkah 3 - Pilih target, perluas Pengaturan tambahan.
3. Untuk Konfigurasi input target, pilih Input transformator di dropdown.

   Klik Konfigurasi transformator input.

   EventBridge menampilkan kotak dialog Configure input transformator.
4. Di bagian Contoh peristiwa, pilih jenis acara Contoh yang ingin Anda uji pola acara Anda. Anda dapat memilih AWS acara, acara mitra, atau memasukkan acara khusus Anda sendiri.

   AWS events

   Pilih dari peristiwa yang dipancarkan dari yang didukung. Layanan AWS

   1. Pilih AWS acara.
   2. Di bawah Contoh peristiwa, pilih AWS acara yang diinginkan. Acara diselenggarakan oleh AWS layanan.

Saat Anda memilih acara, EventBridge mengisi contoh acara.

Misalnya, jika Anda memilih S3 Object Created, EventBridge menampilkan contoh acara S3 Object Created.

3. (Opsional) Anda juga dapat memilih Salin untuk menyalin contoh peristiwa ke clipboard perangkat Anda.

Partner events

Pilih dari peristiwa yang dipancarkan dari layanan pihak ketiga yang mendukung EventBridge, seperti Salesforce.

1. Pilih acara EventBridge mitra.
2. Di bawah Contoh acara, pilih acara mitra yang diinginkan. Acara diselenggarakan oleh mitra.

   Saat Anda memilih acara, EventBridge mengisi contoh acara.
3. (Opsional) Anda juga dapat memilih Salin untuk menyalin contoh peristiwa ke clipboard perangkat Anda.

Enter your own

Masukkan acara Anda sendiri dalam teks JSON.

1. Pilih Masukkan milik Anda sendiri.
2. EventBridge mengisi acara sampel dengan template atribut acara yang diperlukan.
3. Edit dan tambahkan ke acara sampel sesuai keinginan. Contoh acara harus JSON yang valid.
4. (Opsional) Anda juga dapat memilih salah satu opsi berikut:
   - Salin - Salin contoh acara ke clipboard perangkat Anda.
   - Prettify - Membuat teks JSON lebih mudah dibaca dengan menambahkan jeda baris, tab, dan spasi.

5. (Opsional) Perluas jalur masukan Contoh, Template dan Output bagian untuk melihat contoh:

- Bagaimana jalur JSON digunakan untuk mendefinisikan variabel yang mewakili data peristiwa

- Bagaimana variabel-variabel tersebut dapat digunakan dalam template transformator input
- Output yang dihasilkan yang EventBridge mengirim ke target

Untuk contoh transformasi input yang lebih rinci, lihat???.

6. Di bagian Transformator input target, tentukan variabel apa pun yang ingin Anda gunakan dalam template input.

   Variabel menggunakan jalur JSON untuk mereferensikan nilai di sumber acara asli. Anda kemudian dapat mereferensikan variabel-variabel tersebut dalam template input untuk memasukkan data dari peristiwa sumber asli dalam peristiwa yang diubah yang EventBridge lolos ke target. Anda dapat menentukan hingga 100 variabel. Trafo input harus JSON yang valid.

   Misalnya, Anda telah memilih AWS acara S3 Object Created sebagai contoh acara Anda untuk transformator input ini. Anda kemudian dapat menentukan variabel berikut untuk digunakan dalam template Anda:

   ```
   {
     "requester": "$.detail.requester",
     "key": "$.detail.object.key",
     "bucket": "$.detail.bucket.name"
   }
   ```

   (Opsional) Anda juga dapat memilih Salin untuk menyalin transformator input ke clipboard perangkat Anda.

7. Di bagian Template, tulis template yang ingin Anda gunakan untuk menentukan apa yang EventBridge lolos ke target.

   Anda dapat menggunakan JSON, string, informasi statis, variabel yang telah Anda definisikan serta variabel cadangan. Untuk contoh transformasi input yang lebih rinci, lihat???.

   Misalnya, Anda telah mendefinisikan variabel dalam contoh sebelumnya. Anda kemudian dapat membuat template berikut, yang mereferensikan variabel-variabel tersebut, serta variabel cadangan, dan informasi statis.

   ```
   {
       "message": "<requester> has created the object \"<key>\" in the bucket
    \"<bucket>\"",
       "RuleName": <aws.events.rule-name>,
   ```

```
    "ruleArn" : <aws.events.rule-arn>,
    "Transformed": "Yes"
}
```

(Opsional) Anda juga dapat memilih Salin untuk menyalin template ke clipboard perangkat Anda.

8. Untuk menguji template Anda, pilih Hasilkan output.

   EventBridge memproses peristiwa sampel berdasarkan template input, dan menampilkan output yang diubah yang dihasilkan di bawah Output. Ini adalah informasi yang EventBridge akan diteruskan ke target di tempat acara sumber asli.

   Output yang dihasilkan untuk contoh template input yang dijelaskan di atas adalah sebagai berikut:

```
{
    "message": "123456789012 has created the object "example-key" in the bucket
 "example-bucket"",
    "RuleName": rule-name,
    "ruleArn" : arn:aws:events:us-east-1:123456789012:rule/rule-name,
    "Transformed": "Yes"
}
```

   (Opsional) Anda juga dapat memilih Salin untuk menyalin output yang dihasilkan ke clipboard perangkat Anda.

9. Pilih Konfirmasi
10. Ikuti langkah-langkah lainnya untuk membuat aturan seperti yang dijelaskan dalam???.

Menguji transformator input target menggunakan EventBridge Sandbox

Anda dapat menggunakan transformator input untuk menyesuaikan teks dari suatu peristiwa sebelum EventBridge meneruskan informasi ke target aturan.

Mengkonfigurasi transformator input biasanya merupakan bagian dari proses yang lebih besar dalam menentukan target sambil membuat aturan baru atau mengedit yang sudah ada. Namun EventBridge, dengan menggunakan Sandbox, Anda dapat dengan cepat mengonfigurasi transformator input dan menggunakan contoh peristiwa untuk mengonfirmasi bahwa Anda mendapatkan output yang diinginkan, tanpa harus membuat atau mengedit aturan.

Untuk informasi selengkapnya tentang transformasi input, lihat???.

Untuk menguji transformator input target

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di bawah Sumber daya Pengembang, pilih Sandbox, dan pada halaman Sandbox pilih tab Transformator input target.
3. Di bagian Contoh peristiwa, pilih jenis acara Contoh yang ingin Anda uji pola acara Anda. Anda dapat memilih AWS acara, acara mitra, atau memasukkan acara khusus Anda sendiri.

   AWS events

   Pilih dari peristiwa yang dipancarkan dari yang didukung. Layanan AWS

   1. Pilih AWS acara.
   2. Di bawah Contoh peristiwa, pilih AWS acara yang diinginkan. Acara diselenggarakan oleh AWS layanan.

      Saat Anda memilih acara, EventBridge mengisi contoh acara.

      Misalnya, jika Anda memilih S3 Object Created, EventBridge menampilkan contoh acara S3 Object Created.
   3. (Opsional) Anda juga dapat memilih Salin untuk menyalin contoh peristiwa ke clipboard perangkat Anda.

   Partner events

   Pilih dari peristiwa yang dipancarkan dari layanan pihak ketiga yang mendukung EventBridge, seperti Salesforce.

   1. Pilih acara EventBridge mitra.
   2. Di bawah Contoh acara, pilih acara mitra yang diinginkan. Acara diselenggarakan oleh mitra.

      Saat Anda memilih acara, EventBridge mengisi contoh acara.
   3. (Opsional) Anda juga dapat memilih Salin untuk menyalin contoh peristiwa ke clipboard perangkat Anda.

Enter your own

Masukkan acara Anda sendiri dalam teks JSON.

1. Pilih Masukkan milik Anda sendiri.
2. EventBridge mengisi acara sampel dengan template atribut acara yang diperlukan.
3. Edit dan tambahkan ke acara sampel sesuai keinginan. Contoh acara harus JSON yang valid.
4. (Opsional) Anda juga dapat memilih salah satu opsi berikut:
   - Salin - Salin contoh acara ke clipboard perangkat Anda.
   - Prettify - Membuat teks JSON lebih mudah dibaca dengan menambahkan jeda baris, tab, dan spasi.

4. (Opsional) Perluas jalur masukan Contoh, Template dan Output bagian untuk melihat contoh:

   - Bagaimana jalur JSON digunakan untuk mendefinisikan variabel yang mewakili data peristiwa
   - Bagaimana variabel-variabel tersebut dapat digunakan dalam template transformator input
   - Output yang dihasilkan yang EventBridge mengirim ke target

   Untuk contoh transformasi input yang lebih rinci, lihat???.

5. Di bagian Transformator input target, tentukan variabel apa pun yang ingin Anda gunakan dalam template input.

   Variabel menggunakan jalur JSON untuk mereferensikan nilai di sumber acara asli. Anda kemudian dapat mereferensikan variabel-variabel tersebut dalam template input untuk memasukkan data dari peristiwa sumber asli dalam peristiwa yang diubah yang EventBridge lolos ke target. Anda dapat menentukan hingga 100 variabel. Trafo input harus JSON yang valid.

   Misalnya, Anda telah memilih AWS acara S3 Object Created sebagai contoh acara Anda untuk transformator input ini. Anda kemudian dapat menentukan variabel berikut untuk digunakan dalam template Anda:

```
{
  "requester": "$.detail.requester",
  "key": "$.detail.object.key",
  "bucket": "$.detail.bucket.name"
```

```
}
```

(Opsional) Anda juga dapat memilih Salin untuk menyalin transformator input ke clipboard perangkat Anda.

6. Di bagian Template, tulis template yang ingin Anda gunakan untuk menentukan apa yang EventBridge lolos ke target.

   Anda dapat menggunakan JSON, string, informasi statis, variabel yang telah Anda definisikan serta variabel cadangan. Untuk contoh transformasi input yang lebih rinci, lihat???.

   Misalnya, Anda telah mendefinisikan variabel dalam contoh sebelumnya. Anda kemudian dapat membuat template berikut, yang mereferensikan variabel-variabel tersebut, serta variabel cadangan, dan informasi statis.

   ```
   {
       "message": "<requester> has created the object \"<key>\" in the bucket
    \"<bucket>\"",
       "RuleName": <aws.events.rule-name>,
       "ruleArn" : <aws.events.rule-arn>,
       "Transformed": "Yes"
   }
   ```

   (Opsional) Anda juga dapat memilih Salin untuk menyalin template ke clipboard perangkat Anda.

7. Untuk menguji template Anda, pilih Hasilkan output.

   EventBridge memproses peristiwa sampel berdasarkan template input, dan menampilkan output yang diubah yang dihasilkan di bawah Output. Ini adalah informasi yang EventBridge akan diteruskan ke target di tempat acara sumber asli.

   Output yang dihasilkan untuk contoh template input yang dijelaskan di atas adalah sebagai berikut:

   ```
   {
       "message": "123456789012 has created the object "example-key" in the bucket
    "example-bucket"",
       "RuleName": rule-name,
       "ruleArn" : arn:aws:events:us-east-1:123456789012:rule/rule-name,
       "Transformed": "Yes"
   }
   ```

(Opsional) Anda juga dapat memilih Salin untuk menyalin output yang dihasilkan ke clipboard perangkat Anda.

Amazon EventBridge arsip dan putar ulang

Dalam EventBridge, Anda dapat membuat arsip peristiwa sehingga Anda dapat memutar ulang dengan mudah di lain waktu. Misalnya, Anda mungkin ingin memutar ulang peristiwa untuk melakukan pemulihan dari kesalahan atau membutuhkan validasi fungsi baru aplikasi Anda.

> **Note**
>
> Mungkin ada penundaan antara acara yang dipublikasikan ke bus acara dan acara yang tiba di arsip. Sebaiknya Anda menunda memutar ulang acara yang diarsipkan selama 10 menit untuk memastikan semua acara diputar ulang.

Video berikut menunjukkan penggunaan arsip dan replay: Membuat arsip dan replay

Topik

- Mengarsipkan Amazon EventBridge acara
- Memutar ulang EventBridge peristiwa Amazon yang diarsipkan

Mengarsipkan Amazon EventBridge acara

Saat Anda membuat arsip EventBridge, Anda dapat menentukanacaradikirim ke arsip dengan menentukanPola Peristiwa. EventBridge mengirimkan peristiwa yang sesuai dengan pola peristiwa untuk arsip. Anda juga mengatur periode penyimpanan untuk menyimpan peristiwa di arsip sebelum dihapus.

Secara default, EventBridge mengenkripsi data peristiwa dalam arsip menggunakan Advanced Encryption Standard 256-bitAWSCMK yang dimiliki, yang membantu mengamankan data Anda dari akses yang tidak sah.

> **Note**
>
> Acara kadaluarsa biasanya dikurangkan dari`EventCount`dan`SizeBytes`nilai`DescribeArchive`operasi setiap 24 jam. Oleh karena itu, peristiwa yang baru-baru ini kedaluwarsa mungkin tidak tercermin dalam nilai-nilai ini.

Untuk membuat arsip untuk semua peristiwa

1. Buka Amazon EventBridge konsolhttps://console.aws.amazon.com/events/.
2. Pada panel navigasi kiri, pilih Arsip.
3. Pilih Buat arsip.
4. Di bawah Detail arsip, masukkan Nama untuk arsip. Nama harus unik untuk akun Anda di Wilayah yang dipilih.

   Anda tidak dapat mengubah nama setelah Anda membuat arsip.
5. (Opsional) Masukkan Deskripsi untuk arsip.
6. Untuk Sumber, pilih bus peristiwa yang menghasilkan peristiwa untuk dikirim ke arsip.
7. Untuk Periode penyimpanan, lakukan salah satu langkah berikut:
   - Pilih Tak terbatas untuk mempertahankan peristiwa dalam arsip dan tidak pernah menghapusnya.
   - Masukkan jumlah hari untuk mempertahankan acara. Setelah jumlah hari yang ditentukan, EventBridge menghapus peristiwa dari arsip.
8. Pilih Selanjutnya.
9. Di bawah Pola peristiwa, pilih Tidak ada penyaringan peristiwa.

10. Pilih Buat arsip.

Untuk membuat arsip dengan pola peristiwa

1. Buka Amazon EventBridge konsolhttps://console.aws.amazon.com/events/.
2. Pada panel navigasi kiri, pilih Arsip.
3. Pilih Buat arsip.
4. Di bawah Detail arsip, masukkan Nama untuk arsip. Nama harus unik untuk akun Anda di Wilayah yang dipilih.

   Anda tidak dapat mengubah nama setelah Anda membuat arsip.
5. (Opsional) Masukkan Deskripsi untuk arsip.
6. Untuk Sumber, pilih bus peristiwa yang menghasilkan peristiwa untuk dikirim ke arsip.
7. Untuk periode Penyimpanan, lakukan salah satu hal berikut:
   - Pilih Tak terbatas* untuk mempertahankan peristiwa dalam arsip dan tidak pernah menghapusnya.
   - Masukkan jumlah hari untuk mempertahankan acara. Setelah jumlah hari yang ditentukan, EventBridge menghapus peristiwa dari arsip.
8. Pilih Selanjutnya.
9. Di bawah Pola peristiwa, pilih Menyaring peristiwa berdasarkan pencocokan pola acara.
10. Lakukan salah satu dari berikut ini:
    - Pilih Builder pola, lalu pilih Penyedia layanan. Jika Anda memilih AWS, juga pilih AWS nama layanan dan Jenis acara untuk digunakan dalam pola.
    - Pilih Editor JSON untuk membuat pola secara manual. Anda juga dapat menyalin pola dari aturan kemudian menyisipkannya ke editor JSON.
11. Pilih Buat arsip.

Untuk mengonfirmasi bahwa peristiwa berhasil dikirim ke arsip, Anda dapat menggunakan`DescribeArchive`operasi EventBridge API untuk melihat apakah`EventCount`mencerminkan jumlah peristiwa dalam arsip. Jika 0, tidak ada peristiwa dalam arsip.

Memutar ulang EventBridge peristiwa Amazon yang diarsipkan

Setelah membuat arsip, Anda dapat memutar ulang peristiwa dari arsip. Contohnya, jika Anda memperbarui aplikasi dengan fungsionalitas tambahan, Anda dapat memutar ulang peristiwa sejarah untuk memastikan bahwa peristiwa diproses ulang untuk menjaga aplikasi tetap konsisten. Anda juga dapat menggunakan arsip untuk memutar ulang peristiwa untuk fungsionalitas baru. Ketika Anda memutar ulang peristiwa, Anda dapat menentukan arsip mana yang akan digunakan untuk memutar ulang peristiwa dari, mulai dan akhir waktu peristiwa untuk diputar ulang, bus peristiwa, atau satu atau lebih aturan untuk memutar ulang peristiwa.

Peristiwa tidak harus diputar ulang dengan urutan yang sama seperti yang ditambahkan ke arsip. Pemutaran ulang memproses peristiwa untuk diputar ulang berdasarkan waktu peristiwa, dan memutar ulangnya dalam jeda satu menit. Jika Anda menentukan waktu mulai dan waktu selesai peristiwa yang memiliki rentang waktu 20 menit, peristiwa diputar ulang dari menit pertama dari rentang 20 menit pertama. Kemudian peristiwa dari menit kedua diputar ulang. Anda dapat menggunakan`DescribeReplay` operasi EventBridge API untuk menentukan progres pemutaran ulang. `EventLastReplayedTime`mengembalikan cap waktu dari acara terakhir diputar ulang.

Peristiwa diputar ulang berdasarkan, tetapi terpisah dari, `PutEvents` transaksi per batas kedua untuk AWS akun. Anda dapat meminta perpanjangan batas untuk PutEvents Untuk informasi selengkapnya, lihat EventBridge Kuota Amazon.

> **Note**
>
> Anda dapat memiliki maksimal 10 pemutaran ulang yang aktif secara bersamaan per akun per AWS Wilayah.

Untuk mulai pemutaran ulang peristiwa

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Pada panel navigasi kiri, pilih Pemutaran Ulang.
3. Pilih Mulai replay baru.
4. Masukkan Nama untuk pemutaran ulang, dan secara opsional, Deskripsi
5. Untuk Sumber, pilih arsip untuk memutar ulang peristiwa.
6. Untuk tujuan, Anda dapat memutar ulang peristiwa hanya untuk bus peristiwa yang mengeluarkan peristiwa.

7. Untuk Tentukan aturan, lakukan salah satu dari hal berikut:
   - Pilih Semua aturan untuk memutar ulang peristiwa semua aturan.
   - Pilih Tentukan aturan, lalu pilih aturan atau aturan-aturan untuk memutar ulang peristiwa.
8. Dalam Rangka waktu putar ulang, tentukan Tanggal, Waktu, dan Zona waktu untuk Waktu mulai dan Waktu selesai. Hanya peristiwa yang terjadi antara Waktu selesai akan diputar ulang.
9. Pilih Mulai putar ulang.

Ketika peristiwa yang diarsipkan diputar ulang, status pemutaran ulang adalah Selesai.

Jika Anda memulai pemutaran ulang dan ingin memotongnya, Anda dapat membatalkannya selama status Mulai atau Berjalan.

Untuk membatalkan pemutaran ulang

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Pada panel navigasi kiri, pilih Pemutaran Ulang.
3. Pilih putar ulang untuk membatalkan.
4. Pilih Batalkan.

EventBridge Pipa Amazon

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. Ini mengurangi kebutuhan akan pengetahuan khusus dan kode integrasi saat mengembangkan arsitektur berbasis peristiwa, mendorong konsistensi di seluruh aplikasi perusahaan Anda. Untuk menyiapkan pipa, Anda memilih sumber, menambahkan pemfilteran opsional, menentukan pengayaan opsional, dan memilih target untuk data peristiwa.

> **Note**
>
> Anda juga dapat merutekan acara menggunakan bus acara. Bus acara sangat cocok untuk many-to-many perutean acara antara layanan yang digerakkan oleh acara. Untuk informasi selengkapnya, lihat ???.

Bagaimana EventBridge Pipa bekerja

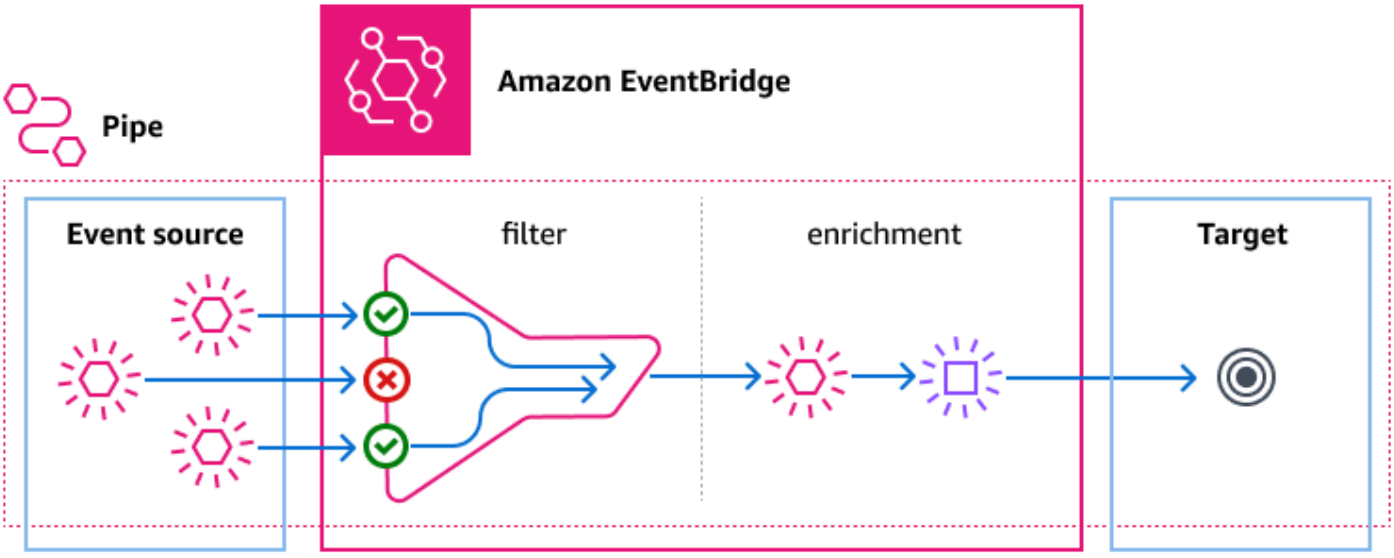
Pada tingkat tinggi, inilah cara kerja EventBridge Pipes:

1. Anda membuat pipa di akun Anda. Hal ini mencakup:
   - Menentukan salah satu sumber acara yang didukung dari mana Anda ingin pipa Anda menerima acara.
   - Secara opsional, mengkonfigurasi filter sehingga pipa hanya memproses subset dari peristiwa yang diterimanya dari sumbernya.
   - Secara opsional, mengonfigurasi langkah pengayaan yang meningkatkan data peristiwa sebelum mengirimkannya ke target.
   - Menentukan salah satu target yang didukung yang Anda inginkan pipa Anda untuk mengirim acara.
2. Sumber acara mulai mengirim peristiwa ke pipa, dan pipa memproses peristiwa sebelum mengirimnya ke target.
   - Jika Anda telah mengonfigurasi filter, pipa mengevaluasi peristiwa dan hanya mengirimkannya ke target jika cocok dengan filter itu.

     Anda hanya dikenakan biaya untuk acara yang cocok dengan filter.

- Jika Anda telah mengonfigurasi pengayaan, pipa melakukan pengayaan itu pada acara tersebut sebelum mengirimkannya ke target.

  Jika acara dikelompokkan, pengayaan mempertahankan urutan acara dalam batch.



Misalnya, pipa dapat digunakan untuk membuat sistem e-commerce. Misalkan Anda memiliki API yang berisi informasi pelanggan, seperti alamat pengiriman.

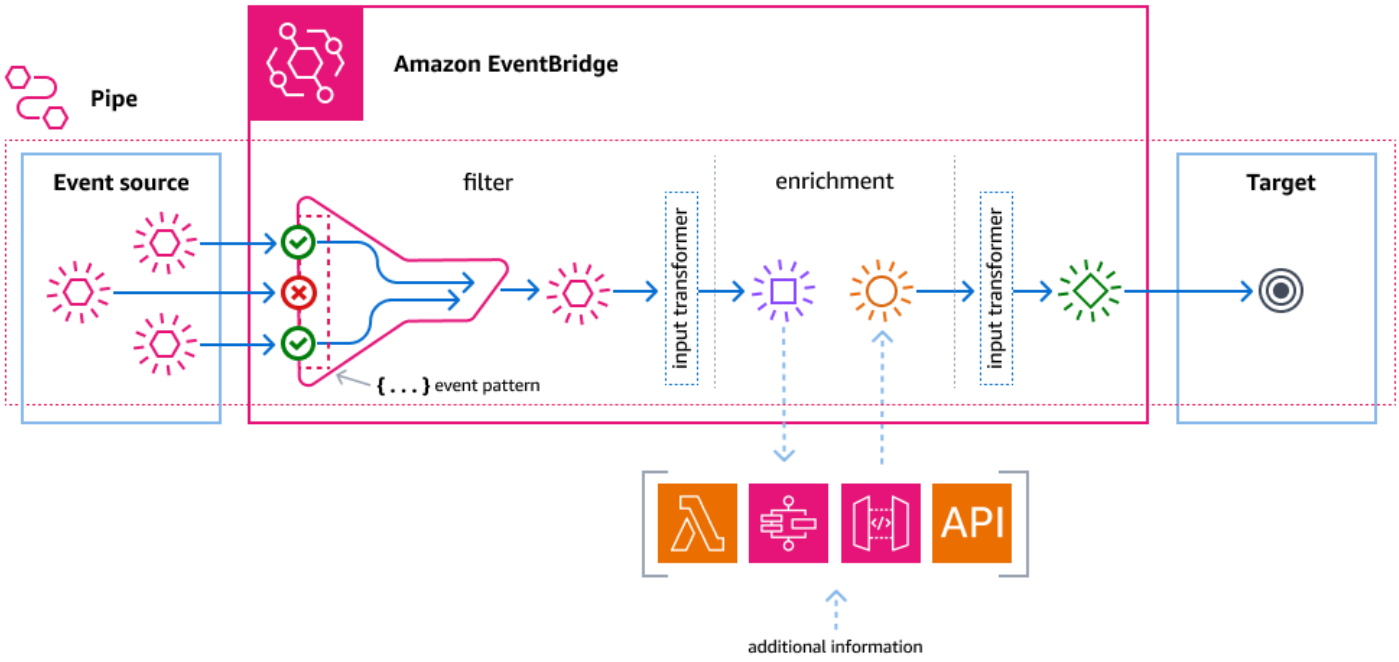
1. Anda kemudian membuat pipa dengan yang berikut:
   - Pesanan Amazon SQS menerima antrian pesan sebagai sumber acara.
   - Tujuan EventBridge API sebagai pengayaan
   - Mesin AWS Step Functions negara sebagai target
2. Kemudian, ketika pesan yang diterima pesanan Amazon SQS muncul di antrian, pesan tersebut dikirim ke pipa Anda.
3. Pipa kemudian mengirimkan data tersebut ke pengayaan Tujuan EventBridge API, yang mengembalikan informasi pelanggan untuk pesanan tersebut.
4. Terakhir, pipa mengirimkan data yang diperkaya ke mesin AWS Step Functions negara, yang memproses pesanan.

EventBridge Konsep pipa

Berikut ini adalah melihat lebih dekat komponen dasar EventBridge Pipa.

Pipa

Pipa merutekan peristiwa dari satu sumber ke satu target. Pipa juga mencakup kemampuan untuk memfilter untuk peristiwa tertentu, dan untuk melakukan pengayaan pada data peristiwa sebelum dikirim ke target.



Sumber

EventBridge Pipes menerima data peristiwa dari berbagai sumber, menerapkan filter opsional dan pengayaan untuk data tersebut, dan mengirimkannya ke target. Jika sumber memberlakukan perintah untuk peristiwa yang dikirim ke pipa, pesanan itu dipertahankan sepanjang seluruh proses ke target.

Untuk informasi selengkapnya tentang sumber, lihat ???.

Filter

Pipa dapat memfilter peristiwa sumber tertentu dan kemudian memproses hanya sebagian dari peristiwa tersebut. Untuk mengonfigurasi pemfilteran pada pipa, Anda menentukan pola peristiwa yang digunakan pipa untuk menentukan peristiwa mana yang akan dikirim ke target.

Anda hanya dikenakan biaya untuk acara yang cocok dengan filter.

Untuk informasi selengkapnya, lihat ???.

Pengayaan

Dengan langkah pengayaan EventBridge Pipa, Anda dapat meningkatkan data dari sumber sebelum mengirimkannya ke target. Misalnya, Anda mungkin menerima acara yang dibuat Tiket yang tidak menyertakan data tiket lengkap. Menggunakan pengayaan, Anda dapat meminta fungsi Lambda memanggil `get-ticket` API untuk detail tiket lengkap. Pipa kemudian dapat mengirim informasi itu ke target.

Untuk informasi selengkapnya tentang memperkaya data peristiwa, lihat. ???

Target

Setelah data peristiwa difilter dan diperkaya, Anda dapat menentukan pipa mengirimkannya ke target tertentu, seperti aliran Amazon Kinesis atau grup log Amazon CloudWatch . Untuk daftar target yang tersedia, lihat???.

Anda dapat mengubah data setelah ditingkatkan dan sebelum dikirim oleh pipa ke target. Untuk informasi selengkapnya, lihat ???.

Beberapa pipa, masing-masing dengan sumber yang berbeda, dapat mengirim acara ke target yang sama.

Anda juga dapat menggunakan pipa dan bus acara bersama-sama untuk mengirim acara ke beberapa target. Kasus penggunaan yang umum adalah membuat pipa dengan bus acara sebagai targetnya; pipa mengirimkan peristiwa ke bus acara, yang kemudian mengirimkan peristiwa tersebut ke beberapa target. Misalnya, Anda dapat membuat pipa dengan aliran DynamoDB untuk sumber, dan bus acara sebagai target. Pipa menerima peristiwa dari aliran DynamoDB dan mengirimkannya ke bus acara, yang kemudian mengirimkannya ke beberapa target sesuai dengan aturan yang telah Anda tentukan di bus acara.

Izin untuk Pipa Amazon EventBridge

Saat mengatur pipa, Anda dapat menggunakan peran eksekusi yang ada, atau EventBridge membuatnya untuk Anda dengan izin yang diperlukan. Izin EventBridge Pipa membutuhkan

bervariasi berdasarkan jenis sumber, dan tercantum di bawah ini. Jika Anda menyiapkan peran eksekusi Anda sendiri, Anda harus menambahkan izin ini sendiri.

> **Note**
>
> Jika Anda tidak yakin izin dengan cakupan yang tepat yang diperlukan untuk mengakses sumber, gunakan konsol EventBridge Pipes untuk membuat peran baru, lalu periksa tindakan yang tercantum dalam kebijakan.

Topik

- Izin peran eksekusi DynamoDB
- Izin peran eksekusi Kinesis
- Izin peran eksekusi Amazon MQ
- Izin peran eksekusi MSK Amazon
- Izin peran eksekusi Apache Kafka yang dikelola sendiri
- Izin peran eksekusi Amazon SQS
- Pengayaan dan izin target

Izin peran eksekusi DynamoDB

Untuk DynamoDB Streams EventBridge , Pipes memerlukan izin berikut untuk mengelola sumber daya yang terkait dengan aliran data DynamoDB Anda.

- `dynamodb:DescribeStream`
- `dynamodb:GetRecords`
- `dynamodb:GetShardIterator`
- `dynamodb:ListStreams`

Untuk mengirim catatan batch yang gagal ke antrean huruf mati pipa, peran eksekusi pipa Anda memerlukan izin berikut:

- `sqs:SendMessage`

Izin peran eksekusi Kinesis

Untuk Kinesis, EventBridge Pipes memerlukan izin berikut untuk mengelola sumber daya yang terkait dengan aliran data Kinesis Anda.

- `kinesis:DescribeStream`
- `kinesis:DescribeStreamSummary`
- `kinesis:GetRecords`
- `kinesis:GetShardIterator`
- `kinesis:ListShards`
- `kinesis:ListStreams`
- `kinesis:SubscribeToShard`

Untuk mengirim catatan batch yang gagal ke antrean huruf mati pipa, peran eksekusi pipa Anda memerlukan izin berikut:

- `sqs:SendMessage`

Izin peran eksekusi Amazon MQ

Untuk Amazon MQ, EventBridge Pipes memerlukan izin berikut untuk mengelola sumber daya yang terkait dengan broker pesan Amazon MQ Anda.

- `mq:DescribeBroker`
- `secretsmanager:GetSecretValue`
- `ec2:CreateNetworkInterface`
- `ec2:DeleteNetworkInterface`
- `ec2:DescribeNetworkInterfaces`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeSubnets`
- `ec2:DescribeVpcs`
- `logs:CreateLogGroup`
- `logs:CreateLogStream`

- logs:PutLogEvents

Izin peran eksekusi MSK Amazon

Untuk Amazon MSK, EventBridge memerlukan izin berikut untuk mengelola sumber daya yang terkait dengan topik MSK Amazon Anda.

> **Note**
>
> Jika Anda menggunakan autentikasi berbasis peran IAM, peran eksekusi Anda akan memerlukan izin yang tercantum di samping yang tercantum ??? di bawah ini.

- kafka:DescribeClusterV2
- kafka:GetBootstrapBrokers
- ec2:CreateNetworkInterface
- ec2:DescribeNetworkInterfaces
- ec2:DescribeVpcs
- ec2:DeleteNetworkInterface
- ec2:DescribeSubnets
- ec2:DescribeSecurityGroups
- logs:CreateLogGroup
- logs:CreateLogStream
- logs:PutLogEvents

Izin peran eksekusi Apache Kafka yang dikelola sendiri

Untuk Apache Kafka yang dikelola sendiri, EventBridge memerlukan izin berikut untuk mengelola sumber daya yang terkait dengan aliran Apache Kafka yang dikelola sendiri.

Izin yang diperlukan

Untuk membuat dan menyimpan log dalam grup log di Amazon CloudWatch Logs, pipa Anda harus memiliki izin berikut dalam peran pelaksanaannya:

- logs:CreateLogGroup
- logs:CreateLogStream
- logs:PutLogEvents

Izin opsional

Pipa Anda mungkin juga memerlukan izin untuk:

- Jelaskan rahasia Secrets Manager Anda.
- Akses AWS Key Management Service (AWS KMS) kunci terkelola pelanggan Anda.
- Akses VPC Amazon Anda.

Secrets Manager dan izin AWS KMS

Bergantung pada jenis kontrol akses yang Anda konfigurasikan untuk broker Apache Kafka Anda, pipa Anda mungkin memerlukan izin untuk mengakses rahasia Secrets Manager Anda atau untuk mendekripsi kunci yang dikelola pelanggan Anda. AWS KMS Untuk mengakses sumber daya ini, peran eksekusi fungsi Anda harus memiliki izin berikut:

- secretsmanager:GetSecretValue
- kms:Decrypt

Izin VPC

Jika hanya pengguna dalam VPC yang dapat mengakses cluster Apache Kafka yang dikelola sendiri, pipa Anda harus memiliki izin untuk mengakses sumber daya Amazon VPC Anda. Sumber daya ini termasuk VPC, subnet, grup keamanan, dan antarmuka jaringan. Untuk mengakses sumber daya ini, peran eksekusi pipa Anda harus memiliki izin berikut:

- ec2:CreateNetworkInterface
- ec2:DescribeNetworkInterfaces
- ec2:DescribeVpcs
- ec2:DeleteNetworkInterface
- ec2:DescribeSubnets

- `ec2:DescribeSecurityGroups`

Izin peran eksekusi Amazon SQS

Untuk Amazon SQS, EventBridge memerlukan izin berikut untuk mengelola sumber daya yang terkait dengan antrian Amazon SQS Anda.

- `sqs:ReceiveMessage`
- `sqs:DeleteMessage`
- `sqs:GetQueueAttributes`

Pengayaan dan izin target

Untuk melakukan panggilan API pada sumber daya yang Anda miliki, EventBridge Pipes memerlukan izin yang sesuai. EventBridge Pipa menggunakan peran IAM yang Anda tentukan pada pipa untuk pengayaan dan panggilan target menggunakan prinsip IAM. `pipes.amazonaws.com`

Membuat EventBridge pipa Amazon

EventBridge Pipes memungkinkan Anda membuat point-to-point integrasi antara sumber dan target, termasuk transformasi dan pengayaan acara lanjutan. Untuk membuat EventBridge pipa, Anda melakukan langkah-langkah berikut:

1. ???
2. ???
3. ???
4. ???
5. ???

Untuk informasi tentang cara membuat pipa menggunakan CLI, lihat create-pipe di AWS CLI Command Reference.AWS

Menentukan sumber

Untuk memulai, tentukan sumber dari mana Anda ingin pipa menerima acara.

Untuk menentukan sumber pipa dengan menggunakan konsol

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Pada panel navigasi, pilih Pipa.
3. Pilih Buat pipa.
4. Masukkan nama untuk pipa.
5. (Opsional) Tambahkan deskripsi untuk pipa.
6. Pada tab Build pipe, untuk Source, pilih jenis sumber yang ingin Anda tentukan untuk pipa ini, dan konfigurasikan sumbernya.

   Properti konfigurasi berbeda berdasarkan jenis sumber yang Anda pilih:

   Confluent

   Untuk mengonfigurasi aliran Confluent Cloud sebagai sumber, dengan menggunakan konsol

   1. Untuk Sumber, pilih Confluent Cloud.
   2. Untuk server Bootstrap, masukkan alamat `host:port` pasangan broker Anda.
   3. Untuk nama Topik, masukkan nama topik yang akan dibaca pipa.
   4. (Opsional) Untuk VPC, pilih VPC yang Anda inginkan. Kemudian, untuk subnet VPC, pilih subnet yang diinginkan. Untuk grup keamanan VPC, pilih grup keamanan.
   5. Untuk Otentikasi - opsional, aktifkan Gunakan Otentikasi dan lakukan hal berikut:
      a. Untuk metode otentikasi, pilih jenis otentikasi.
      b. Untuk kunci Rahasia, pilih kunci rahasia.

      Untuk informasi selengkapnya, lihat Mengautentikasi ke sumber daya Confluent Cloud di dokumentasi Confluent.
   6. (Opsional) Untuk pengaturan tambahan - opsional, lakukan hal berikut:
      a. Untuk posisi awal, pilih salah satu dari berikut ini:
         - Terbaru - Mulailah membaca aliran dengan catatan terbaru di pecahan.
         - Potong cakrawala - Mulailah membaca aliran dengan catatan terakhir yang belum dipangkas di pecahan. Ini adalah rekor tertua di pecahan.
      b. Untuk ukuran Batch - opsional, masukkan jumlah maksimum catatan untuk setiap batch. Nilai default-nya adalah 100.

c. Untuk jendela Batch - opsional, masukkan jumlah maksimum detik untuk mengumpulkan catatan sebelum melanjutkan.

DynamoDB

1. Untuk Sumber, pilih DynamoDB.
2. Untuk aliran DynamoDB, pilih aliran yang ingin Anda gunakan sebagai sumber.
3. Untuk posisi awal, pilih salah satu dari berikut ini:
   - Terbaru - Mulailah membaca aliran dengan catatan terbaru di pecahan.
   - Potong cakrawala - Mulailah membaca aliran dengan catatan terakhir yang belum dipangkas di pecahan. Ini adalah rekor tertua di pecahan.
4. (Opsional) Untuk pengaturan tambahan - opsional, lakukan hal berikut:
   a. Untuk ukuran Batch - opsional, masukkan jumlah maksimum catatan untuk setiap batch. Nilai default-nya adalah 100.
   b. Untuk jendela Batch - opsional, masukkan jumlah maksimum detik untuk mengumpulkan catatan sebelum melanjutkan.
   c. Untuk batch bersamaan per pecahan - opsional, masukkan jumlah batch dari pecahan yang sama yang dapat dibaca pada saat yang sama.
   d. Untuk Kegagalan item batch sebagian, pilih yang berikut ini:
      - AUTOMATIC_BISECT — Potong dua setiap batch dan coba lagi setiap setengahnya sampai semua catatan diproses atau ada satu pesan gagal yang tersisa di batch.

> **Note**
>
> Jika Anda tidak memilih AUTOMATIC_BISECT, Anda dapat mengembalikan catatan gagal tertentu dan hanya yang dicoba ulang.

Kinesis

Untuk mengkonfigurasi sumber Kinesis dengan menggunakan konsol

1. Untuk Sumber, pilih Kinesis.
2. Untuk aliran Kinesis, pilih aliran yang ingin Anda gunakan sebagai sumber.
3. Untuk posisi awal, pilih salah satu dari berikut ini:

- Terbaru - Mulailah membaca aliran dengan catatan terbaru di pecahan.
- Potong cakrawala - Mulailah membaca aliran dengan catatan terakhir yang belum dipangkas di pecahan. Ini adalah rekor tertua di pecahan.
- Pada stempel waktu — Mulai membaca aliran dari waktu yang ditentukan. Di bawah Timestamp, masukkan data dan waktu menggunakan format YYYY/MM/DD dan hh: mm: ss.

4. (Opsional) Untuk pengaturan tambahan - opsional, lakukan hal berikut:
   a. Untuk ukuran Batch - opsional, masukkan jumlah maksimum catatan untuk setiap batch. Nilai default-nya adalah 100.
   b. (Opsional) Untuk jendela Batch - opsional, masukkan jumlah maksimum detik untuk mengumpulkan catatan sebelum melanjutkan.
   c. Untuk batch bersamaan per pecahan - opsional, masukkan jumlah batch dari pecahan yang sama yang dapat dibaca pada saat yang sama.
   d. Untuk Kegagalan item batch sebagian, pilih yang berikut ini:
      - AUTOMATIC_BISECT — Potong dua setiap batch dan coba lagi setiap setengahnya sampai semua catatan diproses atau ada satu pesan gagal yang tersisa di batch.

> **Note**
>
> Jika Anda tidak memilih AUTOMATIC_BISECT, Anda dapat mengembalikan catatan gagal tertentu dan hanya yang dicoba ulang.

Amazon MQ

Untuk mengonfigurasi sumber Amazon MQ dengan menggunakan konsol

1. Untuk Sumber, pilih Amazon MQ.
2. Untuk broker Amazon MQ, pilih aliran yang ingin Anda gunakan sebagai sumber.
3. Untuk nama Antrian, masukkan nama antrian yang akan dibaca pipa.
4. Untuk Metode Otentikasi, pilih BASIC_AUTH.
5. Untuk kunci Rahasia, pilih kunci rahasia.
6. (Opsional) Untuk pengaturan tambahan - opsional, lakukan hal berikut:

a. Untuk ukuran Batch - opsional, masukkan jumlah maksimum pesan untuk setiap batch. Nilai default-nya adalah 100.

b. Untuk jendela Batch - opsional, masukkan jumlah maksimum detik untuk mengumpulkan catatan sebelum melanjutkan.

Amazon MSK

Untuk mengonfigurasi sumber MSK Amazon dengan menggunakan konsol

1. Untuk Sumber, pilih Amazon MSK.
2. Untuk Amazon MSK cluster, pilih cluster yang ingin Anda gunakan.
3. Untuk nama Topik, masukkan nama topik yang akan dibaca pipa.
4. (Opsional) Untuk ID Grup Konsumen - opsional, masukkan ID grup konsumen yang Anda inginkan untuk bergabung dengan pipa.
5. (Opsional) Untuk Otentikasi - opsional, aktifkan Gunakan Otentikasi dan lakukan hal berikut:
   a. Untuk metode Otentikasi, pilih jenis yang Anda inginkan.
   b. Untuk kunci Rahasia, pilih kunci rahasia.
6. (Opsional) Untuk pengaturan tambahan - opsional, lakukan hal berikut:
   a. Untuk ukuran Batch - opsional, masukkan jumlah maksimum catatan untuk setiap batch. Nilai default-nya adalah 100.
   b. Untuk jendela Batch - opsional, masukkan jumlah maksimum detik untuk mengumpulkan catatan sebelum melanjutkan.
   c. Untuk posisi awal, pilih salah satu dari berikut ini:
      - Terbaru - Mulailah membaca topik dengan catatan terbaru di pecahan.
      - Potong cakrawala - Mulailah membaca topik dengan catatan terakhir yang belum dipangkas di pecahan. Ini adalah rekor tertua di pecahan.

> **Note**
>
> Trim horizon sama dengan Earlyest untuk Apache Kafka.

Self managed Apache Kafka

Untuk mengonfigurasi sumber Apache Kafka yang dikelola sendiri dengan menggunakan konsol

1. Untuk Sumber, pilih Apache Kafka yang dikelola sendiri.
2. Untuk server Bootstrap, masukkan alamat `host:port` pasangan broker Anda.
3. Untuk nama Topik, masukkan nama topik yang akan dibaca pipa.
4. (Opsional) Untuk VPC, pilih VPC yang Anda inginkan. Kemudian, untuk subnet VPC, pilih subnet yang diinginkan. Untuk grup keamanan VPC, pilih grup keamanan.
5. (Opsional) Untuk Otentikasi - opsional, aktifkan Gunakan Otentikasi dan lakukan hal berikut:
   a. Untuk metode otentikasi, pilih jenis otentikasi.
   b. Untuk kunci Rahasia, pilih kunci rahasia.
6. (Opsional) Untuk pengaturan tambahan - opsional, lakukan hal berikut:
   a. Untuk posisi awal, pilih salah satu dari berikut ini:
      - Terbaru - Mulailah membaca aliran dengan catatan terbaru di pecahan.
      - Potong cakrawala - Mulailah membaca aliran dengan catatan terakhir yang belum dipangkas di pecahan. Ini adalah rekor tertua di pecahan.
   b. Untuk ukuran Batch - opsional, masukkan jumlah maksimum catatan untuk setiap batch. Nilai default-nya adalah 100.
   c. Untuk jendela Batch - opsional, masukkan jumlah maksimum detik untuk mengumpulkan catatan sebelum melanjutkan.

Amazon SQS

Untuk mengonfigurasi sumber Amazon SQS dengan menggunakan konsol

1. Untuk Sumber, pilih SQS.
2. Untuk antrian SQS, pilih antrian yang ingin Anda gunakan.
3. (Opsional) Untuk pengaturan tambahan - opsional, lakukan hal berikut:
   a. Untuk ukuran Batch - opsional, masukkan jumlah maksimum catatan untuk setiap batch. Nilai default-nya adalah 100.

b. Untuk jendela Batch - opsional, masukkan jumlah maksimum detik untuk mengumpulkan catatan sebelum melanjutkan.

Mengkonfigurasi penyaringan acara (opsional)

Anda dapat menambahkan pemfilteran ke pipa Anda sehingga Anda hanya mengirim sebagian peristiwa dari sumber Anda ke target.

Untuk mengkonfigurasi pemfilteran dengan menggunakan konsol

1. Pilih Penyaringan.
2. Di bawah Contoh acara - opsional, Anda akan melihat contoh peristiwa yang dapat Anda gunakan untuk membangun pola acara Anda, atau Anda dapat memasukkan acara Anda sendiri dengan memilih Enter your own.
3. Di bawah Pola acara, masukkan pola acara yang ingin Anda gunakan untuk memfilter peristiwa. Untuk informasi selengkapnya tentang membuat filter, lihat???.

   Berikut ini adalah contoh pola peristiwa yang hanya mengirimkan peristiwa dengan nilai Seattle di bidang City.

```
{
  "data": {
    "City": ["Seattle"]
  }
}
```

Sekarang peristiwa sedang difilter, Anda dapat menambahkan pengayaan opsional dan target untuk pipa.

Mendefinisikan pengayaan acara (opsional)

Anda dapat mengirim data peristiwa untuk pengayaan ke fungsi Lambda, mesin AWS Step Functions status, Amazon API Gateway, atau tujuan API.

Untuk memilih pengayaan

1. Pilih Pengayaan.

2. Di bawah Detail, untuk Layanan, pilih layanan dan setelan terkait yang ingin Anda gunakan untuk pengayaan.

Anda juga dapat mengubah data sebelum mengirimnya untuk ditingkatkan.

(Opsional) Untuk menentukan transformator input

1. Pilih Enrichment Input Transformer - opsional.
2. Untuk Acara Sampel/Payload Acara, pilih jenis acara sampel.
3. Untuk Transformer, masukkan sintaks transformator, seperti `"Event happened at <$.detail.field>."` di mana `<$.detail.field>` adalah referensi ke bidang dari peristiwa sampel. Anda juga dapat mengklik dua kali bidang dari peristiwa sampel untuk menambahkannya ke transformator.
4. Untuk Output, verifikasi bahwa output terlihat seperti yang Anda inginkan.

Sekarang data telah disaring dan ditingkatkan, Anda harus menentukan target untuk mengirim data acara ke.

Mengkonfigurasi target

Untuk mengkonfigurasi target

1. Pilih Target.
2. Di bawah Detail, untuk layanan Target, pilih target. Bidang yang ditampilkan bervariasi tergantung pada target yang Anda pilih. Masukkan informasi khusus untuk jenis target ini, sesuai kebutuhan.

Anda juga dapat mengubah data sebelum mengirimnya ke target.

(Opsional) Untuk menentukan transformator input

1. Pilih Target Input Transformer - opsional.
2. Untuk Acara Sampel/Payload Acara, pilih jenis acara sampel.
3. Untuk Transformer, masukkan sintaks transformator, seperti `"Event happened at <$.detail.field>."` di mana `<$.detail.field>` adalah referensi ke bidang dari

peristiwa sampel. Anda juga dapat mengklik dua kali bidang dari peristiwa sampel untuk menambahkannya ke transformator.

4. Untuk Output, verifikasi bahwa output terlihat seperti yang Anda inginkan.

Sekarang pipa dikonfigurasi, pastikan pengaturannya dikonfigurasi dengan benar.

Mengkonfigurasi pengaturan pipa

Pipa aktif secara default, tetapi Anda dapat menonaktifkannya. Anda juga dapat menentukan izin pipa, mengatur logging pipa, dan menambahkan tag.

Untuk mengkonfigurasi pengaturan pipa

1. Pilih tab Pengaturan pipa.
2. Secara default, pipa yang baru dibuat aktif segera setelah dibuat. Jika Anda ingin membuat pipa tidak aktif, di bawah Aktivasi, untuk Aktifkan pipa, matikan Aktif.
3. Di bawah Izin, untuk peran Eksekusi, lakukan salah satu hal berikut:

   a. Untuk EventBridge membuat peran eksekusi baru untuk pipa ini, pilih Buat peran baru untuk sumber daya khusus ini. Di bawah Nama peran, Anda dapat mengedit nama peran secara opsional.

   b. Untuk menggunakan peran eksekusi yang ada, pilih Gunakan peran yang ada. Di bawah Nama peran, pilih peran.
4. (Opsional) Jika Anda telah menentukan DynamoDB aliran Kinesis atau sebagai sumber pipa, Anda dapat mengonfigurasi kebijakan coba lagi dan antrian huruf mati (DLQ).

   Untuk kebijakan Coba lagi dan antrian Dead-letter - opsional, lakukan hal berikut:

   Di bawah kebijakan Coba Ulang, lakukan hal berikut:

   a. Jika Anda ingin mengaktifkan kebijakan coba lagi, aktifkan Coba lagi. Secara default, pipa yang baru dibuat tidak mengaktifkan kebijakan coba lagi.

   b. Untuk Masa peristiwa maksimal, masukkan nilai antara satu menit (00:01) dan 24 jam (24:00).

   c. Untuk Upaya coba lagi, masukkan angka antara 0 dan 185.

d. Jika Anda ingin menggunakan antrian huruf mati (DLQ), aktifkan antrian Dead-letter, pilih metode pilihan Anda, dan pilih antrian atau topik yang ingin Anda gunakan. Secara default, pipa yang baru dibuat tidak menggunakan DLQ.

5. (Opsional) Di bawah Log - opsional, Anda dapat mengatur cara EventBridge Pipes mengirimkan informasi pencatatan ke layanan yang didukung, termasuk cara mengonfigurasi log tersebut.

   Untuk informasi selengkapnya tentang pencatatan pipa logging, lihat???.

   CloudWatch log dipilih sebagai tujuan log secara default, seperti tingkat `ERROR` log. Jadi, secara default, EventBridge Pipes membuat grup CloudWatch log baru yang mengirimkan catatan log yang berisi `ERROR` tingkat detail.

   Agar EventBridge Pipes mengirim catatan log ke salah satu tujuan log yang didukung, lakukan hal berikut:

   a. Di bawah Log - opsional, pilih tujuan yang ingin Anda kirimkan catatan log.

   b. Untuk tingkat Log, pilih tingkat informasi EventBridge untuk disertakan dalam catatan log. Tingkat `ERROR` log dipilih secara default.

      Untuk informasi selengkapnya, lihat ???.

   c. Pilih Sertakan data eksekusi jika Anda EventBridge ingin menyertakan informasi muatan peristiwa dan permintaan layanan dan informasi respons dalam catatan log.

      Untuk informasi selengkapnya, lihat ???.

   d. Konfigurasikan setiap tujuan log yang Anda pilih:

      Untuk CloudWatch Logs log, di bawah CloudWatch log lakukan hal berikut:

      - Untuk grup CloudWatch log, pilih apakah akan EventBridge membuat grup log baru, atau Anda dapat memilih grup log yang ada atau menentukan ARN dari grup log yang ada.
      - Untuk grup log baru, edit nama grup log sesuai keinginan.

      CloudWatch log dipilih secara default.

      Untuk log Firehose aliran, di bawah log Firehose aliran, pilih Firehose aliran.

      Untuk Amazon S3 log, di bawah log S3 lakukan hal berikut:

      - Masukkan nama bucket untuk digunakan sebagai tujuan log.

- Masukkan ID AWS akun pemilik bucket.
- Masukkan teks awalan yang ingin Anda gunakan saat EventBridge membuat objek S3.

  Untuk informasi selengkapnya, lihat Mengatur objek menggunakan awalan di Amazon Simple Storage Service Panduan Pengguna.
- Pilih bagaimana Anda ingin EventBridge memformat catatan log S3:
  - `json`: JSON
  - `plain`: Teks biasa
  - `w3c`: Format file logging diperpanjang W3C

6. (Opsional) Di bawah Tag - opsional, pilih Tambahkan tag baru dan masukkan satu atau beberapa tag untuk aturan. Untuk informasi selengkapnya, lihat ???.
7. Pilih Buat pipa.

Memvalidasi parameter konfigurasi

Setelah pipa dibuat, EventBridge memvalidasi parameter konfigurasi berikut:

- Peran IAM — Karena sumber pipa tidak dapat diubah setelah pipa dibuat, EventBridge memverifikasi bahwa peran IAM yang disediakan dapat mengakses sumbernya.

> **Note**
>
> EventBridge tidak melakukan validasi yang sama untuk pengayaan atau target karena dapat diperbarui setelah pipa dibuat.

- Batching — EventBridge memvalidasi bahwa ukuran batch sumber tidak melebihi ukuran batch maksimum target. Jika ya, EventBridge membutuhkan ukuran batch yang lebih rendah. Selain itu, jika target tidak mendukung batching, Anda tidak dapat mengonfigurasi batching EventBridge untuk sumbernya.
- Pengayaan — EventBridge memvalidasi bahwa ukuran batch untuk API Gateway dan pengayaan tujuan API adalah 1 karena hanya ukuran batch 1 yang didukung.

Memulai atau menghentikan pipa

Secara default, pipa adalah`Running` dan memproses peristiwa saat dibuat.

Jika Anda membuat pipa dengan sumber Amazon SQS, Kinesis, atau DynamoDB, pembuatan pipa biasanya dapat memakan waktu satu atau dua menit.

Jika Anda membuat pipa dengan Amazon MSK, Apache Kafka yang dikelola sendiri, atau sumber Amazon MQ, pembuatan pipa dapat memakan waktu hingga sepuluh menit.

Untuk membuat pipa tanpa memproses acara menggunakan konsol

- Matikan pengaturan pipa Aktifkan.

Untuk membuat pipa tanpa memproses peristiwa pemrograman

- Dalam panggilan API Anda, atur`DesiredState` ke`Stopped`.

Untuk memulai atau menghentikan pipa yang ada menggunakan konsol

- Pada tab Pengaturan pipa, di bawah Aktivasi, untuk Aktifkan pipa, aktifkan atau nonaktifkan Aktif.

Untuk memulai atau menghentikan pipa yang ada

- Dalam panggilan API Anda, atur`DesiredState` parameter ke salah satu`RUNNING` atau`STOPPED`.

Mungkin ada penundaan antara kapan pipa berada`STOPPED` dan ketika tidak lagi memproses peristiwa:

- Untuk Amazon SQS dan sumber streaming, penundaan ini biasanya kurang dari dua menit.
- Untuk sumber Amazon MQ dan Apache Kafka, penundaan ini mungkin sampai lima belas menit.

Sumber EventBridge Pipa Amazon

EventBridge Pipes menerima data peristiwa dari berbagai sumber, menerapkan filter opsional dan pengayaan untuk data tersebut, dan mengirimkannya ke tujuan.

Jika sumber memberlakukan perintah untuk peristiwa yang dikirim ke EventBridge Pipes, pesanan itu dipertahankan selama seluruh proses ke tujuan.

AWS Layanan berikut dapat ditentukan sebagai sumber untuk EventBridge Pipa:

- Aliran Amazon DynamoDB
- Aliran Amazon Kinesis
- Pialang Amazon MQ
- Aliran MSK Amazon
- Antrian Amazon SQS
- Aliran Apache Kafka

  Saat Anda menentukan aliran Apache Kafka sebagai sumber pipa, Anda dapat menentukan aliran Apache Kafka yang Anda kelola sendiri, atau yang dikelola oleh penyedia pihak ketiga seperti:

  - Confluent Cloud
  - CloudKarafka
  - Redpanda

Amazon DynamoDB stream sebagai sumber

Anda dapat menggunakanEventBridgePipa untuk menerima catatan dalam aliran DynamoDB. Anda kemudian dapat secara opsional memfilter atau menyempurnakan catatan ini sebelum mengirimnya ke target untuk diproses. Ada pengaturan khusus untuk Amazon DynamoDB Streams yang dapat Anda pilih saat mengatur pipa. EventBridgePipes menjaga urutan catatan dari aliran data saat mengirim data tersebut ke tujuan.

Important

Menonaktifkan aliran DynamoDB yang merupakan sumber pipa mengakibatkan pipa itu menjadi tidak dapat digunakan, bahkan jika Anda kemudian mengaktifkan kembali aliran tersebut. Ini terjadi karena:

- Anda tidak dapat menghentikan, memulai, atau memperbarui pipa yang sumbernya dinonaktifkan.
- Anda tidak dapat memperbarui pipa dengan sumber baru setelah pembuatan. Saat Anda mengaktifkan kembali aliran DynamoDB, aliran tersebut diberi Amazon Resource Name (ARN) baru, dan tidak lagi terkait dengan pipa Anda.

Jika Anda mengaktifkan kembali aliran DynamoDB, Anda kemudian perlu membuat pipa baru menggunakan ARN baru aliran.

Contoh acara

Contoh peristiwa berikut menunjukkan informasi yang diterima oleh pipa. Anda dapat menggunakan acara ini untuk membuat dan memfilter pola acara Anda, atau untuk menentukan transformasi input. Tidak semua bidang dapat disaring. Untuk informasi selengkapnya tentang bidang mana yang dapat Anda filter, lihat???.


```
[
  {
    "eventID": "1",
    "eventVersion": "1.0",
    "dynamodb": {
      "Keys": {
        "Id": {
          "N": "101"
        }
      },
      "NewImage": {
        "Message": {
          "S": "New item!"
        },
        "Id": {
          "N": "101"
        }
      },
      "StreamViewType": "NEW_AND_OLD_IMAGES",
      "SequenceNumber": "111",
      "SizeBytes": 26
    },
    "awsRegion": "us-west-2",
    "eventName": "INSERT",
    "eventSourceARN": "arn:aws:dynamodb:us-east-1:111122223333:table/EventSourceTable",
    "eventSource": "aws:dynamodb"
  },
  {
    "eventID": "2",
    "eventVersion": "1.0",
    "dynamodb": {
      "OldImage": {
        "Message": {
          "S": "New item!"
        },
        "Id": {
```

```
          "N": "101"
        }
      },
      "SequenceNumber": "222",
      "Keys": {
        "Id": {
          "N": "101"
        }
      },
      "SizeBytes": 59,
      "NewImage": {
        "Message": {
          "S": "This item has changed"
        },
        "Id": {
          "N": "101"
        }
      },
      "StreamViewType": "NEW_AND_OLD_IMAGES"
    },
    "awsRegion": "us-west-2",
    "eventName": "MODIFY",
    "eventSourceARN": "arn:aws:dynamodb:us-east-1:111122223333:table/EventSourceTable",
    "eventSource": "aws:dynamodb"
  }
]
```

Polling dan batching stream

EventBridgepolling pecahan dalam aliran DynamoDB Anda untuk catatan pada tingkat dasar empat kali per detik. Ketika catatan tersedia,EventBridgememproses acara dan menunggu hasilnya. Jika pemrosesan berhasil,EventBridgemelanjutkan pemungutan suara sampai menerima lebih banyak catatan.

Secara default,EventBridgememanggil pipa Anda segera setelah catatan tersedia. Jika batch ituEventBridgebacaan dari sumber hanya memiliki satu catatan di dalamnya, hanya satu peristiwa yang diproses. Untuk menghindari pemrosesan sejumlah kecil catatan, Anda dapat memberi tahu pipa untuk menyangga catatan hingga lima menit dengan mengonfigurasi jendela batching. Sebelum memproses acara,EventBridgeterus membaca catatan dari sumber hingga mengumpulkan batch penuh, jendela batching kedaluwarsa, atau batch mencapai batas muatan 6 MB.

Anda juga dapat meningkatkan konkurensi dengan memproses beberapa batch dari setiap shard secara paralel. EventBridgedapat memproses hingga 10 batch di setiap pecahan secara bersamaan. Jika Anda menambah jumlah batch bersamaan per pecahan,EventBridgemasih memastikan pemrosesan in-order pada tingkat kunci partisi.

KonfigurasikanParallelizationFactorpengaturan untuk memproses satu pecahan aliran data Kinesis atau DynamoDB dengan lebih dari satu eksekusi Pipa secara bersamaan. Anda dapat menentukan jumlah batch bersamaan yangEventBridgejajak pendapat dari pecahan melalui faktor paralelisasi dari 1 (default) ke 10. Misalnya, ketika Anda mengaturParallelizationFactorke 2, Anda dapat memiliki 200 bersamaanEventBridgeEksekusi pipa maksimal untuk memproses 100 pecahan data Kinesis. Hal ini membantu meningkatkan skala throughput pemrosesan ketika volume data tidak stabil dan IteratorAge tinggi. Perhatikan bahwa faktor paralelisasi tidak akan berfungsi jika Anda menggunakan agregasi Kinesis.

Posisi awal polling dan streaming

Ketahuilah bahwa polling sumber aliran selama pembuatan dan pembaruan pipa pada akhirnya konsisten.

- Selama pembuatan pipa, mungkin perlu beberapa menit untuk memulai acara pemungutan suara dari aliran.
- Selama pembaruan pipa ke konfigurasi pemungutan suara sumber, mungkin diperlukan beberapa menit untuk menghentikan dan memulai kembali acara pemungutan suara dari aliran.

Ini berarti bahwa jika Anda menentukanLATESTsebagai posisi awal untuk aliran, pipa dapat melewatkan acara yang dikirim selama pembuatan atau pembaruan pipa. Untuk memastikan tidak ada peristiwa yang terlewatkan, tentukan posisi awal aliran sebagaiTRIM_HORIZON.

Melaporkan kegagalan item batch

KapanEventBridgemengkonsumsi dan memproses streaming data dari suatu sumber, secara default itu memeriksa pos ke nomor urutan tertinggi dari batch, tetapi hanya ketika batch berhasil total. Untuk menghindari pemrosesan ulang pesan yang berhasil diproses dalam kumpulan yang gagal, Anda dapat mengonfigurasi pengayaan atau target untuk mengembalikan objek yang menunjukkan pesan mana yang berhasil dan mana yang gagal. Ini disebut respon batch paral.

Untuk informasi selengkapnya, lihat ???.

Status berhasil dan gagal

Jika Anda mengembalikan salah satu dari yang berikut ini,EventBridgememperlakukan batch sebagai kesuksesan total:

- Daftar `batchItemFailure` kosong
- Daftar `batchItemFailure` nol
- `EventResponse` kosong
- `EventResponse` nol

Jika Anda mengembalikan salah satu dari yang berikut ini,EventBridgememperlakukan batch sebagai kegagalan total:

- String `itemIdentifier` kosong
- `itemIdentifier` nol
- `itemIdentifier` dengan nama kunci yang buruk

EventBridgemencoba kembali kegagalan berdasarkan strategi coba lagi Anda.

Aliran Amazon Kinesis sebagai sumber

Anda dapat menggunakanEventBridgePipa untuk menerima catatan dalam aliran data Kinesis. Anda kemudian dapat secara opsional memfilter atau menyempurnakan catatan ini sebelum mengirimnya ke salah satu tujuan yang tersedia untuk diproses. Ada pengaturan khusus untuk Kinesis yang dapat Anda pilih saat mengatur pipa. EventBridgePipes menjaga urutan catatan dari aliran data saat mengirim data tersebut ke tujuan.

Aliran data Kinesis adalah serangkaian shard. Setiap shard berisi urutan rekaman data. Konsumen adalah aplikasi yang memproses data dari aliran data Kinesis. Anda dapat memetakanEventBridgePipa ke konsumen throughput bersama (iterator standar), atau ke konsumen throughput khusus denganfan-out yang ditingkatkan.

Untuk iterator standar,EventBridgemenggunakan protokol HTTP untuk polling setiap pecahan di aliran Kinesis Anda untuk catatan. Pipa berbagi throughput baca dengan konsumen pecahan lainnya.

Untuk meminimalkan latensi dan memaksimalkan throughput baca, Anda dapat membuat konsumen aliran data dengan keluaran yang ditingkatkan. Konsumen aliran mendapatkan koneksi khusus ke setiap shard yang tidak memengaruhi pembacaan aplikasi lain dari aliran tersebut. Throughput

khusus dapat membantu jika Anda memiliki banyak aplikasi yang membaca data yang sama, atau jika Anda memproses ulang aliran dengan rekaman yang besar. Kinesis mendorong catatan keEventBridgemelalui HTTP/2. Untuk informasi tentang aliran data Kinesis, lihatMembaca Data dari Amazon Kinesis Data Streams.

Contoh acara

Contoh peristiwa berikut menunjukkan informasi yang diterima oleh pipa. Anda dapat menggunakan acara ini untuk membuat dan memfilter pola acara Anda, atau untuk menentukan transformasi input. Tidak semua bidang dapat disaring. Untuk informasi selengkapnya tentang bidang mana yang dapat Anda filter, lihat???.


```
[
  {
    "kinesisSchemaVersion": "1.0",
    "partitionKey": "1",
    "sequenceNumber": "49590338271490256608559692538361571095921575989136588898",
    "data": "SGVsbG8sIHRoaXMgaXMgYSB0ZXN0Lg==",
    "approximateArrivalTimestamp": 1545084650.987
    "eventSource": "aws:kinesis",
    "eventVersion": "1.0",
    "eventID":
 "shardId-000000000006:49590338271490256608559692538361571095921575989136588898",
    "eventName": "aws:kinesis:record",
    "invokeIdentityArn": "arn:aws:iam::123456789012:role/lambda-role",
    "awsRegion": "us-east-2",
    "eventSourceARN": "arn:aws:kinesis:us-east-2:123456789012:stream/lambda-stream"
  },
  {
    "kinesisSchemaVersion": "1.0",
    "partitionKey": "1",
    "sequenceNumber": "49590338271490256608559692540925702759324208523137515618",
    "data": "VGhpcyBpcyBvbmx5IGEgdGVzdC4=",
    "approximateArrivalTimestamp": 1545084711.166
    "eventSource": "aws:kinesis",
    "eventVersion": "1.0",
    "eventID":
 "shardId-000000000006:49590338271490256608559692540925702759324208523137515618",
    "eventName": "aws:kinesis:record",
    "invokeIdentityArn": "arn:aws:iam::123456789012:role/lambda-role",
    "awsRegion": "us-east-2",
    "eventSourceARN": "arn:aws:kinesis:us-east-2:123456789012:stream/lambda-stream"
  }
```

```
]
```

Polling dan batching stream

EventBridgepolling pecahan di aliran Kinesis Anda untuk catatan dengan kecepatan dasar empat kali per detik. Ketika catatan tersedia,EventBridgememproses acara dan menunggu hasilnya. Jika pemrosesan berhasil,EventBridgemelanjutkan pemungutan suara sampai menerima lebih banyak catatan.

Secara default,EventBridgememanggil pipa Anda segera setelah catatan tersedia. Jika batch ituEventBridgebacaan dari sumber hanya memiliki satu catatan di dalamnya, hanya satu peristiwa yang diproses. Untuk menghindari pemrosesan sejumlah kecil catatan, Anda dapat memberi tahu pipa untuk menyangga catatan hingga lima menit dengan mengonfigurasi jendela batching. Sebelum memproses acara,EventBridgeterus membaca catatan dari sumber hingga mengumpulkan batch penuh, jendela batching kedaluwarsa, atau batch mencapai batas muatan 6 MB.

Anda juga dapat meningkatkan konkurensi dengan memproses beberapa batch dari setiap shard secara paralel. EventBridgedapat memproses hingga 10 batch di setiap pecahan secara bersamaan. Jika Anda menambah jumlah batch bersamaan per pecahan,EventBridgemasih memastikan pemrosesan in-order pada tingkat kunci partisi.

KonfigurasikanParallelizationFactorpengaturan untuk memproses satu pecahan aliran data Kinesis atau DynamoDB dengan lebih dari satu eksekusi Pipa secara bersamaan. Anda dapat menentukan jumlah batch bersamaan yangEventBridgejajak pendapat dari pecahan melalui faktor paralelisasi dari 1 (default) ke 10. Misalnya, ketika Anda mengaturParallelizationFactorke 2, Anda dapat memiliki 200 bersamaanEventBridgeEksekusi pipa maksimal untuk memproses 100 pecahan data Kinesis. Hal ini membantu meningkatkan skala throughput pemrosesan ketika volume data tidak stabil dan IteratorAge tinggi. Perhatikan bahwa faktor paralelisasi tidak akan berfungsi jika Anda menggunakan agregasi Kinesis.

Posisi awal polling dan streaming

Ketahuilah bahwa polling sumber aliran selama pembuatan dan pembaruan pipa pada akhirnya konsisten.

- Selama pembuatan pipa, mungkin perlu beberapa menit untuk memulai acara pemungutan suara dari aliran.
- Selama pembaruan pipa ke konfigurasi pemungutan suara sumber, mungkin diperlukan beberapa menit untuk menghentikan dan memulai kembali acara pemungutan suara dari aliran.

Ini berarti bahwa jika Anda menentukan`LATEST`sebagai posisi awal untuk aliran, pipa dapat melewatkan acara yang dikirim selama pembuatan atau pembaruan pipa. Untuk memastikan tidak ada peristiwa yang terlewatkan, tentukan posisi awal aliran sebagai`TRIM_HORIZON`atau`AT_TIMESTAMP`.

Melaporkan kegagalan item batch

KapanEventBridgemengkonsumsi dan memproses streaming data dari suatu sumber, secara default itu memeriksa pos ke nomor urutan tertinggi dari batch, tetapi hanya ketika batch berhasil total. Untuk menghindari pemrosesan ulang pesan yang berhasil diproses dalam kumpulan yang gagal, Anda dapat mengonfigurasi pengayaan atau target untuk mengembalikan objek yang menunjukkan pesan mana yang berhasil dan mana yang gagal. Ini disebut respon batch paral.

Untuk informasi selengkapnya, lihat ???.

Status berhasil dan gagal

Jika Anda mengembalikan salah satu dari yang berikut ini,EventBridgememperlakukan batch sebagai kesuksesan total:

- Daftar `batchItemFailure` kosong
- Daftar `batchItemFailure` nol
- `EventResponse` kosong
- `EventResponse` nol

Jika Anda mengembalikan salah satu dari yang berikut ini,EventBridgememperlakukan batch sebagai kegagalan total:

- String `itemIdentifier` kosong
- `itemIdentifier` nol
- `itemIdentifier` dengan nama kunci yang buruk

EventBridgemencoba kembali kegagalan berdasarkan strategi coba lagi Anda.

Broker pesan Amazon MQ sebagai sumber

Anda dapat menggunakan EventBridge Pipes untuk menerima catatan dari broker pesan Amazon MQ. Anda kemudian dapat secara opsional memfilter atau menyempurnakan catatan ini sebelum

mengirimnya ke salah satu tujuan yang tersedia untuk diproses. Ada pengaturan khusus untuk Amazon MQ yang dapat Anda pilih saat menyiapkan pipa. EventBridge Pipes menjaga urutan catatan dari broker pesan saat mengirim data tersebut ke tujuan.

Amazon MQ adalah layanan broker pesan terkelola untuk Apache ActiveMQ dan RabbitMQ. Broker pesan memungkinkan aplikasi dan komponen perangkat lunak untuk berkomunikasi menggunakan bahasa pemrograman yang berbeda, sistem operasi, dan protokol pesan formal dengan topik atau antrian sebagai tujuan acara.

Amazon MQ juga dapat mengelola instans Amazon Elastic Compute Cloud (Amazon EC2) atas nama Anda dengan menginstal broker ActiveMQ atau RabbitMQ. Setelah broker diinstal, ia menyediakan topologi jaringan yang berbeda dan kebutuhan infrastruktur lainnya untuk instans Anda.

Sumber Amazon MQ memiliki batasan konfigurasi berikut:

- Cross account — EventBridge tidak mendukung pemrosesan lintas akun. Anda tidak dapat menggunakan EventBridge untuk memproses catatan dari broker pesan Amazon MQ yang ada di akun berbedaAWS.
- Otentikasi - Untuk ActiveMQ, hanya ActiveMQ yang didukung. SimpleAuthenticationPlugin Untuk RabbitMQ, hanya mekanisme otentikasi PLAIN yang didukung. Untuk mengelola kredensyal, gunakan. AWS Secrets Manager Untuk informasi selengkapnya tentang otentikasi ActiveMQ, lihat Mengintegrasikan broker ActiveMQ dengan LDAP di Panduan Pengembang Amazon MQ.
- Kuota koneksi — Broker memiliki jumlah maksimum koneksi yang diizinkan untuk setiap protokol tingkat kabel. Kuota ini didasarkan pada jenis instans broker. Untuk informasi selengkapnya, lihat bagian Broker*Kuota di Amazon MQ* di Panduan Pengembang Amazon MQ.
- Konektivitas — Anda dapat membuat broker di cloud pribadi virtual publik atau pribadi (VPC). Untuk VPC pribadi, pipa Anda memerlukan akses ke VPC untuk menerima pesan.
- Tujuan acara - Hanya tujuan antrian yang didukung. Namun, Anda dapat menggunakan topik virtual, yang berperilaku baik sebagai topik internal maupun sebagai antrian eksternal ketika berinteraksi dengan pipa Anda. Untuk informasi lebih lanjut, lihat Tujuan Virtual di situs web Apache ActiveMQ, dan Host Virtual di situs web RabbitMQ.
- Topologi jaringan - Untuk ActiveMQ, hanya satu broker tunggal atau siaga yang didukung untuk pipa. Untuk RabbitMQ, hanya satu broker instans tunggal atau penerapan cluster yang didukung untuk setiap pipa. Broker instans tunggal memerlukan titik akhir failover. Untuk informasi selengkapnya tentang mode penyebaran broker ini, lihat Arsitektur Broker MQ Aktif dan Arsitektur Broker MQ Kelinci di Panduan Pengembang Amazon MQ.
- Protokol — Protokol yang didukung bergantung pada integrasi Amazon MQ yang Anda gunakan.

- Untuk integrasi ActiveMQ EventBridge , gunakan protokol /Java Message Service (JMS) untuk menggunakan OpenWire pesan. Konsumsi pesan tidak didukung pada protokol lain. EventBridge hanya mendukung TextMessagedan BytesMessageoperasi dalam protokol JMS. Untuk informasi lebih lanjut tentang OpenWire protokol, lihat OpenWiredi situs web Apache ActiveMQ.
- Untuk integrasi RabbitMQ, EventBridge gunakan protokol AMQP 0-9-1 untuk menggunakan pesan. Tidak ada protokol lain yang didukung untuk mengkonsumsi pesan. Untuk informasi lebih lanjut tentang implementasi protokol AMQP 0-9-1 RabbitMQ, lihat AMQP 0-9-1 Panduan Referensi Lengkap di situs web RabbitMQ.

EventBridge secara otomatis mendukung versi terbaru ActiveMQ dan RabbitMQ yang didukung Amazon MQ. Untuk versi terbaru yang didukung, lihat catatan rilis Amazon MQ di Panduan Pengembang Amazon MQ.

> **Note**
>
> Secara default, Amazon MQ memiliki jangka waktu pemeliharaan mingguan untuk broker. Selama jangka waktu tersebut, broker tidak tersedia. Untuk broker tanpa siaga, EventBridge tidak akan memproses pesan sampai jendela berakhir.

Contoh peristiwa

Contoh peristiwa berikut menunjukkan informasi yang diterima oleh pipa. Anda dapat menggunakan acara ini untuk membuat dan memfilter pola acara Anda, atau untuk menentukan transformasi input. Tidak semua bidang dapat disaring. Untuk informasi selengkapnya tentang bidang mana yang dapat Anda filter, lihat???.

ActiveMQ

```
[
  {
    "eventSource": "aws:amq",
    "eventSourceArn": "arn:aws:mq:us-
west-2:112556298976:broker:test:b-9bcfa592-423a-4942-879d-eb284b418fc8",
    "messageID": "ID:b-9bcfa592-423a-4942-879d-eb284b418fc8-1.mq.us-
west-2.amazonaws.com-37557-1234520418293-4:1:1:1:1",
    "messageType": "jms/text-message",
    "data": "QUJDOkFBQUE=",
    "connectionId": "myJMSCoID",
```

```
    "redelivered": false,
    "destination": {
      "physicalname": "testQueue"
    },
    "timestamp": 1598827811958,
    "brokerInTime": 1598827811958,
    "brokerOutTime": 1598827811959
  },
  {
    "eventSource": "aws:amq",
    "eventSourceArn": "arn:aws:mq:us-
west-2:112556298976:broker:test:b-9bcfa592-423a-4942-879d-eb284b418fc8",
    "messageID": "ID:b-9bcfa592-423a-4942-879d-eb284b418fc8-1.mq.us-
west-2.amazonaws.com-37557-1234520418293-4:1:1:1:1",
    "messageType": "jms/bytes-message",
    "data": "3DTOOW7crj51prgVLQaGQ82S48k=",
    "connectionId": "myJMSCoID1",
    "persistent": false,
    "destination": {
      "physicalname": "testQueue"
    },
    "timestamp": 1598827811958,
    "brokerInTime": 1598827811958,
    "brokerOutTime": 1598827811959
  }
]
```

RabbitMQ

```
[
  {
    "eventSource": "aws:rmq",
    "eventSourceArn": "arn:aws:mq:us-
west-2:111122223333:broker:pizzaBroker:b-9bcfa592-423a-4942-879d-eb284b418fc8",
    "eventSourceKey": "pizzaQueue::/",
    "basicProperties": {
      "contentType": "text/plain",
      "contentEncoding": null,
      "headers": {
        "header1": {
          "bytes": [
            118,
            97,
```

```
            108,
            117,
            101,
            49
          ]
        },
        "header2": {
          "bytes": [
            118,
            97,
            108,
            117,
            101,
            50
          ]
        },
        "numberInHeader": 10
      },
      "deliveryMode": 1,
      "priority": 34,
      "correlationId": null,
      "replyTo": null,
      "expiration": "60000",
      "messageId": null,
      "timestamp": "Jan 1, 1970, 12:33:41 AM",
      "type": null,
      "userId": "AIDACKCEVSQ6C2EXAMPLE",
      "appId": null,
      "clusterId": null,
      "bodySize": 80
    },
    "redelivered": false,
    "data": "eyJ0aW1lb3V0IjowLCJkYXRhIjoiQ1pybWYwR3c4T3Y0YnFMUXhENEUifQ=="
  }
]
```

Kelompok konsumen

Untuk berinteraksi dengan Amazon MQ, EventBridge buat grup konsumen yang dapat membaca dari broker MQ Amazon Anda. Grup konsumen dibuat dengan ID yang sama dengan pipa UUID.

Untuk sumber Amazon MQ, mengumpulkan EventBridge catatan bersama dan mengirimkannya ke fungsi Anda dalam satu muatan. Untuk mengontrol perilaku, Anda dapat mengonfigurasi jendela batching dan ukuran batch. EventBridge menarik pesan sampai salah satu hal berikut terjadi:

- Catatan yang diproses mencapai ukuran muatan maksimum 6 MB.
- Jendela batching kedaluwarsa.
- Jumlah catatan mencapai ukuran batch penuh.

EventBridge mengubah batch Anda menjadi satu payload dan kemudian memanggil fungsi Anda. Pesan tidak bertahan atau dideserialisasi. Sebaliknya, grup konsumen mengambilnya sebagai BLOB byte. Kemudian base64-mengkodekannya ke dalam muatan JSON. Jika pipa mengembalikan kesalahan untuk salah satu pesan dalam batch, EventBridge coba ulang seluruh kumpulan pesan hingga pemrosesan berhasil atau pesan kedaluwarsa.

Konfigurasi jaringan

Secara default, broker Amazon MQ dibuat dengan flag `PubliclyAccessible` yang ditetapkan ke false. Hanya ketika `PubliclyAccessible` diatur ke true bahwa broker menerima alamat IP publik. Untuk akses penuh dengan pipa Anda, broker Anda harus menggunakan titik akhir publik atau menyediakan akses ke VPC.

Jika broker Amazon MQ Anda tidak dapat diakses publik, EventBridge harus memiliki akses ke sumber daya Amazon Virtual Private Cloud (Amazon VPC) yang terkait dengan broker Anda. Untuk mengakses VPC broker Amazon MQ Anda EventBridge , memerlukan akses internet keluar untuk subnet sumber Anda. Untuk subnet publik ini harus menjadi gateway NAT terkelola. Untuk subnet pribadi itu bisa menjadi gateway NAT, atau NAT Anda sendiri. Pastikan NAT memiliki alamat IP publik dan dapat terhubung ke internet.

Konfigurasikan grup keamanan Amazon VPC Anda dengan aturan berikut (minimal):

- Aturan masuk – Untuk broker tanpa aksesibilitas publik, izinkan semua lalu lintas di semua port untuk grup keamanan yang ditentukan sebagai sumber Anda. Untuk broker dengan aksesibilitas publik, izinkan semua lalu lintas di semua port untuk semua tujuan.
- Aturan keluar – Izinkan semua lalu lintas di semua port untuk semua tujuan.

> **Note**
>
> Konfigurasi VPC Amazon Anda dapat ditemukan melalui Amazon MQ API. Anda tidak perlu mengkonfigurasinya selama pengaturan.

Amazon Managed Streaming untuk topik Apache Kafka sebagai sumber

Anda dapat menggunakan EventBridge Pipes untuk menerima catatan dari topik Amazon Managed Streaming for Apache Kafka (Amazon MSK). Anda dapat secara opsional memfilter atau menyempurnakan catatan ini sebelum mengirimnya ke salah satu tujuan yang tersedia untuk diproses. Ada pengaturan khusus untuk Amazon MSK yang dapat Anda pilih saat menyiapkan pipa. EventBridge Pipes menjaga urutan catatan dari broker pesan saat mengirim data tersebut ke tujuan.

Amazon MSK adalah layanan yang dikelola sepenuhnya yang dapat Anda gunakan untuk membangun dan menjalankan aplikasi yang menggunakan Apache Kafka untuk memproses data streaming. Amazon MSK menyederhanakan pengaturan, penskalaan, dan pengelolaan cluster yang menjalankan Apache Kafka. Dengan Amazon MSK, Anda dapat mengonfigurasi aplikasi Anda untuk beberapa Availability Zone dan untuk keamanan dengan AWS Identity and Access Management (IAM). Amazon MSK mendukung beberapa versi open-source Kafka.

Amazon MSK sebagai sumber beroperasi mirip dengan menggunakan Amazon Simple Queue Service (Amazon SQS) atau Amazon Kinesis. EventBridge polling internal untuk pesan baru dari sumber dan kemudian secara sinkron memanggil target. EventBridge membaca pesan dalam batch dan menyediakannya ke fungsi Anda sebagai muatan acara. Ukuran batch maksimum dapat dikonfigurasi. (Default adalah 100 pesan.)

Untuk sumber berbasis Apache Kafka, EventBridge mendukung parameter kontrol pemrosesan, seperti jendela batching dan ukuran batch.

EventBridge membaca pesan secara berurutan untuk setiap partisi. Setelah EventBridge memproses setiap batch, ia melakukan offset pesan dalam batch itu. Jika target pipa mengembalikan kesalahan untuk salah satu pesan dalam batch, EventBridge coba ulang seluruh kumpulan pesan hingga pemrosesan berhasil atau pesan kedaluwarsa.

EventBridge mengirim kumpulan pesan dalam acara ketika memanggil target. Muatan peristiwa berisi array pesan. Setiap item array berisi detail dari topik Amazon MSK dan pengidentifikasi partisi, bersama-sama dengan stempel waktu dan pesan berkode base64.

Contoh peristiwa

Contoh peristiwa berikut menunjukkan informasi yang diterima oleh pipa. Anda dapat menggunakan acara ini untuk membuat dan memfilter pola acara Anda, atau untuk menentukan transformasi input. Tidak semua bidang dapat disaring. Untuk informasi selengkapnya tentang bidang mana yang dapat Anda filter, lihat???.


```
[
  {
    "eventSource": "aws:kafka",
    "eventSourceArn": "arn:aws:kafka:sa-east-1:123456789012:cluster/
vpc-2priv-2pub/751d2973-a626-431c-9d4e-d7975eb44dd7-2",
    "eventSourceKey": "mytopic-0",
    "topic": "mytopic",
    "partition": "0",
    "offset": 15,
    "timestamp": 1545084650987,
    "timestampType": "CREATE_TIME",
    "key":"abcDEFghiJKLmnoPQRstuVWXyz1234==",
    "value":"SGVsbG8sIHRoaXMgaXMgYSB0ZXN0Lg==",
    "headers": [
      {
        "headerKey": [
          104,
          101,
          97,
          100,
          101,
          114,
          86,
          97,
          108,
          117,
          101
        ]
      }
    ]
  }
]
```


Posisi awal polling dan streaming

Ketahuilah bahwa polling sumber aliran selama pembuatan dan pembaruan pipa pada akhirnya konsisten.

- Selama pembuatan pipa, mungkin perlu beberapa menit untuk memulai acara pemungutan suara dari aliran.
- Selama pembaruan pipa ke konfigurasi pemungutan suara sumber, mungkin diperlukan beberapa menit untuk menghentikan dan memulai kembali acara pemungutan suara dari aliran.

Ini berarti bahwa jika Anda menentukan `LATEST` sebagai posisi awal untuk aliran, pipa dapat melewatkan peristiwa yang dikirim selama pembuatan atau pembaruan pipa. Untuk memastikan tidak ada peristiwa yang terlewatkan, tentukan posisi awal aliran sebagai`TRIM_HORIZON`.

Otentikasi kluster MSK

EventBridge memerlukan izin untuk mengakses kluster MSK Amazon, mengambil catatan, dan melakukan tugas lainnya. Amazon MSK mendukung beberapa opsi untuk mengontrol akses klien ke cluster MSK. Untuk informasi selengkapnya tentang metode otentikasi yang digunakan saat, lihat???.

Opsi akses cluster

- Akses tidak diautentikasi
- Otentikasi SASL/SCRAM
- Autentikasi berbasis peran IAM
- Otentikasi TLS timbal balik
- Mengkonfigurasi rahasia mTLS
- Bagaimana EventBridge memilih broker bootstrap

Akses tidak diautentikasi

Kami merekomendasikan hanya menggunakan akses yang tidak diautentikasi untuk pengembangan. Akses yang tidak diautentikasi hanya akan berfungsi jika autentikasi berbasis peran IAM dinonaktifkan untuk cluster.

Otentikasi SASL/SCRAM

Amazon MSK mendukung otentikasi Simple Authentication and Security Layer/Salted Challenge Response Authentication Mechanism (SASL/SCRAM) otentikasi dengan enkripsi Transport Layer Security (TLS). Untuk terhubung EventBridge ke klaster, Anda menyimpan kredensyal otentikasi (kredensyal masuk) secara rahasia. AWS Secrets Manager

Untuk informasi selengkapnya tentang menggunakan Secrets Manager, lihat Autentikasi nama pengguna dan kata sandi dengan AWS Secrets Manager di Panduan Pengembang Amazon Managed Streaming for Apache Kafka.

Amazon MSK tidak mendukung otentikasi SASL/PLAIN.

Autentikasi berbasis peran IAM

Anda dapat menggunakan IAM untuk mengautentikasi identitas klien yang terhubung ke cluster MSK. Jika autentikasi IAM aktif di kluster MSK Anda, dan Anda tidak memberikan rahasia untuk otentikasi, EventBridge secara otomatis default menggunakan otentikasi IAM. Untuk membuat dan menerapkan pengguna IAM atau kebijakan berbasis peran, gunakan konsol IAM atau API. Untuk informasi selengkapnya, lihat Kontrol akses IAM di Panduan Pengembang Amazon Managed Streaming for Apache Kafka Kafka.

Untuk memungkinkan terhubung EventBridge ke kluster MSK, membaca catatan, dan melakukan tindakan lain yang diperlukan, tambahkan izin berikut ke peran eksekusi pipa Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:Connect",
                "kafka-cluster:DescribeGroup",
                "kafka-cluster:AlterGroup",
                "kafka-cluster:DescribeTopic",
                "kafka-cluster:ReadData",
                "kafka-cluster:DescribeClusterDynamicConfiguration"
            ],
            "Resource": [
                "arn:aws:kafka:region:account-id:cluster/cluster-name/cluster-uuid",
                "arn:aws:kafka:region:account-id:topic/cluster-name/cluster-uuid/topic-
name",
                "arn:aws:kafka:region:account-id:group/cluster-name/cluster-
uuid/consumer-group-id"
            ]
        }
    ]
}
```

Anda dapat membuat cakupan izin ini ke klaster, topik, dan grup tertentu. Untuk informasi selengkapnya, lihat tindakan Amazon MSK Kafka di Panduan Pengembang Amazon Managed Streaming for Apache Kafka.

Otentikasi TLS timbal balik

Mutual TLS (mTLS) menyediakan otentikasi dua arah antara klien dan server. Klien mengirimkan sertifikat ke server untuk server untuk memverifikasi klien, dan server mengirimkan sertifikat ke klien untuk klien untuk memverifikasi server.

Untuk Amazon MSK, EventBridge bertindak sebagai klien. Anda mengonfigurasi sertifikat klien (sebagai rahasia di Secrets Manager) untuk mengautentikasi EventBridge dengan broker di klaster MSK Anda. Sertifikat klien harus ditandatangani oleh otoritas sertifikat (CA) di toko kepercayaan server. Kluster MSK mengirimkan sertifikat server untuk EventBridge mengotentikasi broker dengan. EventBridge Sertifikat server harus ditandatangani oleh CA yang ada di toko AWS kepercayaan.

Amazon MSK tidak mendukung sertifikat server yang ditandatangani sendiri, karena semua broker di Amazon MSK menggunakan sertifikat publik yang ditandatangani oleh Amazon Trust Services CA, yang EventBridge dipercaya secara default.

Untuk informasi selengkapnya tentang MTL untuk Amazon MSK, lihat Otentikasi TLS Bersama di Panduan Pengembang Amazon Managed Streaming for Apache Kafka Kafka.

Mengkonfigurasi rahasia mTLS

Rahasia CLIENT_CERTIFICATE_TLS_AUTH memerlukan bidang sertifikat dan bidang kunci pribadi. Untuk kunci pribadi terenkripsi, rahasianya memerlukan kata sandi kunci pribadi. Baik sertifikat dan kunci pribadi harus dalam format PEM.

> **Note**
>
> EventBridge mendukung algoritma enkripsi kunci pribadi PBES1 (tetapi bukan PBES2).

Bidang sertifikat harus berisi daftar sertifikat, dimulai dengan sertifikat klien, diikuti oleh sertifikat perantara, dan diakhiri dengan sertifikat root. Setiap sertifikat harus dimulai pada baris baru dengan struktur berikut:

```
-----BEGIN CERTIFICATE-----
```

```
        <certificate contents>
-----END CERTIFICATE-----
```

Secrets Manager mendukung rahasia hingga 65.536 byte, yang merupakan ruang yang cukup untuk rantai sertifikat yang panjang.

Kunci pribadi harus dalam format PKCS #8, dengan struktur berikut:

```
-----BEGIN PRIVATE KEY-----
         <private key contents>
-----END PRIVATE KEY-----
```

Untuk kunci pribadi terenkripsi, gunakan struktur berikut:

```
-----BEGIN ENCRYPTED PRIVATE KEY-----
          <private key contents>
-----END ENCRYPTED PRIVATE KEY-----
```

Contoh berikut menunjukkan isi rahasia untuk otentikasi mTLS menggunakan kunci pribadi terenkripsi. Untuk kunci pribadi terenkripsi, Anda menyertakan kata sandi kunci pribadi dalam rahasia.

```
{
 "privateKeyPassword": "testpassword",
 "certificate": "-----BEGIN CERTIFICATE-----
MIIE5DCCAsygAwIBAgIRAPJdwaFaNRrytHBto0j5BA0wDQYJKoZIhvcNAQELBQAw
...
j0Lh4/+1HfgyE2KlmII36dg4IMzNjAFEBZiCRoPimO40s1cRqtFHXoal0QQbIlxk
cmUuiAii9R0=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIFgjCCA2qgAwIBAgIQdjNZd6uFf9hbNC5RdfmHrzANBgkqhkiG9w0BAQsFADBb
...
rQoiowbbk5wXCheYSANQIfTZ6weQTgiCHCCbuuMKNVS95FkXm0vqVD/YpXKwA/no
c8PH3PSoAaRwMMgOSA2ALJvbRz8mpg==
-----END CERTIFICATE-----",
 "privateKey": "-----BEGIN ENCRYPTED PRIVATE KEY-----
MIIFKzBVBgkqhkiG9w0BBQ0wSDAnBgkqhkiG9w0BBQwwGgQUiAFcK5hT/X7Kjmgp
...
QrSekqF+kWzmB6nAfSzgO9IaoAaytLvNgGTckWeUkWn/V0Ck+LdGUXzAC4RxZnoQ
zp2mwJn2NYB7AZ7+imp0azDZb+8YG2aUCiyqb6PnnA==
-----END ENCRYPTED PRIVATE KEY-----"
```

```
}
```

Bagaimana EventBridge memilih broker bootstrap

EventBridge memilih broker bootstrap berdasarkan metode otentikasi yang tersedia di cluster Anda, dan apakah Anda memberikan rahasia untuk otentikasi. Jika Anda memberikan rahasia untuk mTLS atau SASL/SCRAM, EventBridge secara otomatis memilih metode otentikasi itu. Jika Anda tidak memberikan rahasia, EventBridge pilih metode otentikasi terkuat yang aktif di klaster Anda. Berikut ini adalah urutan prioritas di mana EventBridge memilih broker, dari otentikasi terkuat hingga terlemah:

- mTL (rahasia disediakan untuk mTL)
- SASL/SCRAM (rahasia disediakan untuk SASL/SCRAM)
- SASL IAM (tidak ada rahasia yang disediakan, dan otentikasi IAM aktif)
- TLS yang tidak diautentikasi (tidak ada rahasia yang disediakan, dan otentikasi IAM tidak aktif)
- Plaintext (tidak ada rahasia yang disediakan, dan otentikasi IAM dan TLS yang tidak diautentikasi tidak aktif)

> **Note**
>
> Jika tidak EventBridge dapat terhubung ke jenis broker yang paling aman, ia tidak mencoba untuk terhubung ke jenis broker yang berbeda (lebih lemah). Jika Anda EventBridge ingin memilih jenis broker yang lebih lemah, nonaktifkan semua metode otentikasi yang lebih kuat di cluster Anda.

Konfigurasi jaringan

EventBridge harus memiliki akses ke sumber daya Amazon Virtual Private Cloud (Amazon VPC) yang terkait dengan kluster MSK Amazon Anda. Untuk mengakses VPC cluster MSK Amazon Anda, EventBridge memerlukan akses internet keluar untuk subnet sumber Anda. Untuk subnet publik ini harus menjadi gateway NAT terkelola. Untuk subnet pribadi itu bisa menjadi gateway NAT, atau NAT Anda sendiri. Pastikan NAT memiliki alamat IP publik dan dapat terhubung ke internet.

Konfigurasikan grup keamanan Amazon VPC Anda dengan aturan berikut (minimal):

- Aturan masuk - Izinkan semua lalu lintas di port broker MSK Amazon (9092 untuk teks biasa, 9094 untuk TLS, 9096 untuk SASL, 9098 untuk IAM) untuk grup keamanan yang ditentukan untuk sumber Anda.

- Aturan keluar - Izinkan semua lalu lintas di port 443 untuk semua tujuan. Izinkan semua lalu lintas di port broker MSK Amazon (9092 untuk teks biasa, 9094 untuk TLS, 9096 untuk SASL, 9098 untuk IAM) untuk grup keamanan yang ditentukan untuk sumber Anda.

> **Note**
>
> Konfigurasi Amazon VPC Anda dapat ditemukan melalui Amazon MSK API. Anda tidak perlu mengkonfigurasinya selama pengaturan.

ID grup konsumen yang dapat disesuaikan

Saat mengatur Apache Kafka sebagai sumber, Anda dapat menentukan ID grup konsumen. ID grup konsumen ini adalah pengenal yang ada untuk grup konsumen Apache Kafka yang Anda inginkan untuk bergabung dengan pipa Anda. Anda dapat menggunakan fitur ini untuk memigrasikan pengaturan pemrosesan catatan Apache Kafka yang sedang berlangsung dari konsumen lain ke. EventBridge

Jika Anda menentukan ID grup konsumen dan ada poller aktif lainnya dalam grup konsumen tersebut, Apache Kafka mendistribusikan pesan ke semua konsumen. Dengan kata lain, EventBridge tidak menerima semua pesan untuk topik Apache Kafka. Jika Anda EventBridge ingin menangani semua pesan dalam topik, matikan poller lain di grup konsumen tersebut.

Selain itu, jika Anda menentukan ID grup konsumen, dan Apache Kafka menemukan grup konsumen yang sudah ada dengan ID yang sama, EventBridge mengabaikan `StartingPosition` parameter untuk pipa Anda. Sebaliknya, EventBridge mulailah memproses catatan sesuai dengan offset yang dilakukan dari kelompok konsumen. Jika Anda menentukan ID grup konsumen, dan Apache Kafka tidak dapat menemukan grup konsumen yang ada, maka EventBridge konfigurasikan sumber Anda dengan yang ditentukan. `StartingPosition`

ID grup konsumen yang Anda tentukan harus unik di antara semua sumber acara Apache Kafka Anda. Setelah membuat pipa dengan ID grup konsumen yang ditentukan, Anda tidak dapat memperbarui nilai ini.

Penskalaan otomatis dari sumber MSK Amazon

Saat Anda awalnya membuat sumber MSK Amazon, EventBridge alokasikan satu konsumen untuk memproses semua partisi dalam topik Apache Kafka. Setiap konsumen memiliki beberapa prosesor

yang berjalan secara paralel untuk menangani peningkatan beban kerja. Selain itu, EventBridge secara otomatis meningkatkan atau menurunkan jumlah konsumen, berdasarkan beban kerja. Untuk mempertahankan pemesanan pesan di setiap partisi, jumlah maksimum konsumen adalah satu konsumen per partisi dalam topik.

Dalam interval satu menit, EventBridge mengevaluasi lag offset konsumen dari semua partisi dalam topik. Jika lag terlalu tinggi, partisi menerima pesan lebih cepat daripada yang EventBridge dapat memprosesnya. Jika perlu, EventBridge menambah atau menghapus konsumen dari topik. Proses penskalaan penambahan atau penghapusan konsumen terjadi dalam waktu tiga menit setelah evaluasi.

Jika target Anda kelebihan beban, EventBridge kurangi jumlah konsumen. Tindakan ini mengurangi beban kerja pada pipa dengan mengurangi jumlah pesan yang dapat diambil dan dikirim konsumen ke pipa.

Apache Kafka mengalir sebagai sumber

Apache Kafka adalah platform streaming acara open-source yang mendukung beban kerja seperti saluran data dan analitik streaming. Anda dapat menggunakan Amazon Managed Streaming for Apache Kafka (Amazon MSK), atau cluster Apache Kafka yang dikelola sendiri. Dalam AWS terminologi, cluster yang dikelola sendiri mengacu pada cluster Apache Kafka yang tidak di-host oleh. AWS Ini termasuk kedua cluster yang Anda kelola sendiri, serta yang dihosting oleh penyedia pihak ketiga, seperti Confluent Cloud, CloudKarafka, atau Redpanda.

Untuk informasi selengkapnya tentang opsi AWS hosting lain untuk klaster Anda, lihat Praktik Terbaik untuk Menjalankan Apache Kafka AWS di Blog AWS Big Data.

Apache Kafka sebagai sumber beroperasi mirip dengan menggunakan Amazon Simple Queue Service (Amazon SQS) atau Amazon Kinesis. EventBridgepolling internal untuk pesan baru dari sumber dan kemudian secara sinkron memanggil target. EventBridge membaca pesan dalam batch dan menyediakannya ke fungsi Anda sebagai muatan acara. Ukuran batch maksimum dapat dikonfigurasi. (Default adalah 100 pesan.)

Untuk sumber berbasis Apache Kafka, EventBridge mendukung parameter kontrol pemrosesan, seperti jendela batching dan ukuran batch.

EventBridge mengirimkan kumpulan pesan dalam parameter acara saat memanggil pipa Anda. Muatan peristiwa berisi array pesan. Setiap item array berisi rincian topik Apache Kafka dan pengidentifikasi partisi Apache Kafka, bersama dengan stempel waktu dan pesan yang dikodekan base64.

Contoh peristiwa

Contoh peristiwa berikut menunjukkan informasi yang diterima oleh pipa. Anda dapat menggunakan acara ini untuk membuat dan memfilter pola acara Anda, atau untuk menentukan transformasi input. Tidak semua bidang dapat disaring. Untuk informasi selengkapnya tentang bidang mana yang dapat Anda filter, lihat???.

```
[
  {
    "eventSource": "SelfManagedKafka",
    "bootstrapServers": "b-2.demo-cluster-1.a1bcde.c1.kafka.us-
east-1.amazonaws.com:9092,b-1.demo-cluster-1.a1bcde.c1.kafka.us-
east-1.amazonaws.com:9092",
    "eventSourceKey": "mytopic-0",
    "topic": "mytopic",
    "partition": 0,
    "offset": 15,
    "timestamp": 1545084650987,
    "timestampType": "CREATE_TIME",
    "key":"abcDEFghiJKLmnoPQRstuVWXyz1234==",
    "value":"SGVsbG8sIHRoaXMgaXMgYSB0ZXN0Lg==",
    "headers": [
      {
        "headerKey": [
          104,
          101,
          97,
          100,
          101,
          114,
          86,
          97,
          108,
          117,
          101
        ]
      }
    ]
  }
]
```

Otentikasi cluster Apache Kafka

EventBridge Pipes mendukung beberapa metode untuk mengotentikasi dengan cluster Apache Kafka yang dikelola sendiri. Pastikan Anda mengonfigurasi cluster Apache Kafka untuk menggunakan salah satu metode otentikasi yang didukung ini. Untuk informasi lebih lanjut tentang keamanan Apache Kafka, lihat bagian Keamanan dari dokumentasi Apache Kafka.

Akses VPC

Jika Anda menggunakan lingkungan Apache Kafka yang dikelola sendiri di mana hanya pengguna Apache Kafka dalam VPC Anda yang memiliki akses ke broker Apache Kafka Anda, Anda harus mengonfigurasi Amazon Virtual Private Cloud (Amazon VPC) di sumber Apache Kafka.

Otentikasi SASL/SCRAM

EventBridge Pipes mendukung otentikasi Simple Authentication and Security Layer/Salted Challenge Response Authentication Mechanism (SASL/SCRAM) otentikasi dengan enkripsi Transport Layer Security (TLS). EventBridge Pipes mengirimkan kredensyal terenkripsi untuk mengautentikasi dengan cluster. Untuk informasi selengkapnya tentang autentikasi SASL/SCRAM, lihat RFC 5802.

EventBridge Pipa mendukung otentikasi SASL/PLAIN dengan enkripsi TLS. Dengan otentikasi SASL/PLAIN, EventBridge Pipes mengirimkan kredensyal sebagai teks yang jelas (tidak terenkripsi) ke server.

Untuk autentikasi SASL, Anda menyimpan kredensyal masuk sebagai rahasia. AWS Secrets Manager

Otentikasi TLS timbal balik

Mutual TLS (mTLS) menyediakan otentikasi dua arah antara klien dan server. Klien mengirimkan sertifikat ke server untuk server untuk memverifikasi klien, dan server mengirimkan sertifikat ke klien untuk klien untuk memverifikasi server.

Dalam Apache Kafka yang dikelola sendiri, EventBridge Pipes bertindak sebagai klien. Anda mengonfigurasi sertifikat klien (sebagai rahasia di Secrets Manager) untuk mengautentikasi EventBridge Pipes dengan broker Apache Kafka Anda. Sertifikat klien harus ditandatangani oleh otoritas sertifikat (CA) di toko kepercayaan server.

Cluster Apache Kafka mengirimkan sertifikat server ke EventBridge Pipes untuk mengautentikasi broker Apache Kafka dengan Pipes. EventBridge Sertifikat server dapat berupa sertifikat CA publik

atau sertifikat CA/Self-signed private. Sertifikat CA publik harus ditandatangani oleh CA yang ada di toko kepercayaan EventBridge Pipes. Untuk sertifikat CA/Self-signed privat, Anda mengonfigurasi sertifikat CA root server (sebagai rahasia di Secrets Manager). EventBridge Pipes menggunakan sertifikat root untuk memverifikasi broker Apache Kafka.

Untuk informasi selengkapnya tentang mTL, lihat Memperkenalkan otentikasi TLS timbal balik untuk Amazon MSK sebagai sumber.

Mengkonfigurasi rahasia sertifikat klien

Rahasia CLIENT_CERTIFICATE_TLS_AUTH memerlukan bidang sertifikat dan bidang kunci pribadi. Untuk kunci pribadi terenkripsi, rahasianya memerlukan kata sandi kunci pribadi. Baik sertifikat dan kunci pribadi harus dalam format PEM.

> **Note**
>
> EventBridge Pipa mendukung algoritma enkripsi kunci pribadi PBES1 (tetapi bukan PBES2).

Bidang sertifikat harus berisi daftar sertifikat, dimulai dengan sertifikat klien, diikuti oleh sertifikat perantara, dan diakhiri dengan sertifikat root. Setiap sertifikat harus dimulai pada baris baru dengan struktur berikut:

```
-----BEGIN CERTIFICATE-----
        <certificate contents>
-----END CERTIFICATE-----
```

Secrets Manager mendukung rahasia hingga 65.536 byte, yang merupakan ruang yang cukup untuk rantai sertifikat yang panjang.

Kunci pribadi harus dalam format PKCS #8, dengan struktur berikut:

```
-----BEGIN PRIVATE KEY-----
         <private key contents>
-----END PRIVATE KEY-----
```

Untuk kunci pribadi terenkripsi, gunakan struktur berikut:

```
-----BEGIN ENCRYPTED PRIVATE KEY-----
          <private key contents>
```

```
-----END ENCRYPTED PRIVATE KEY-----
```

Contoh berikut menunjukkan isi rahasia untuk otentikasi mTLS menggunakan kunci pribadi terenkripsi. Untuk kunci pribadi terenkripsi, sertakan kata sandi kunci pribadi dalam rahasia.

```
{
 "privateKeyPassword": "testpassword",
 "certificate": "-----BEGIN CERTIFICATE-----
MIIE5DCCAsygAwIBAgIRAPJdwaFaNRrytHBto0j5BA0wDQYJKoZIhvcNAQELBQAw
...
j0Lh4/+1HfgyE2KlmII36dg4IMzNjAFEBZiCRoPimO40s1cRqtFHXoal0QQbIlxk
cmUuiAii9R0=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIFgjCCA2qgAwIBAgIQdjNZd6uFf9hbNC5RdfmHrzANBgkqhkiG9w0BAQsFADBb
...
rQoiowbbk5wXCheYSANQIfTZ6weQTgiCHCCbuuMKNVS95FkXm0vqVD/YpXKwA/no
c8PH3PSoAaRwMMgOSA2ALJvbRz8mpg==
-----END CERTIFICATE-----",
 "privateKey": "-----BEGIN ENCRYPTED PRIVATE KEY-----
MIIFKzBVBgkqhkiG9w0BBQ0wSDAnBgkqhkiG9w0BBQwwGgQUiAFcK5hT/X7Kjmgp
...
QrSekqF+kWzmB6nAfSzgO9IaoAaytLvNgGTckWeUkWn/V0Ck+LdGUXzAC4RxZnoQ
zp2mwJn2NYB7AZ7+imp0azDZb+8YG2aUCiyqb6PnnA==
-----END ENCRYPTED PRIVATE KEY-----"
}
```

Mengkonfigurasi rahasia sertifikat CA root server

Anda membuat rahasia ini jika broker Apache Kafka Anda menggunakan enkripsi TLS dengan sertifikat yang ditandatangani oleh CA pribadi. Anda dapat menggunakan enkripsi TLS untuk otentikasi VPC, SASL/SCRAM, SASL/PLAIN, atau mTLS.

Rahasia sertifikat CA root server memerlukan bidang yang berisi sertifikat root CA broker Apache Kafka dalam format PEM. Contoh berikut menunjukkan struktur rahasia.

```
{
     "certificate": "-----BEGIN CERTIFICATE-----
  MIID7zCCAtegAwIBAgIBADANBgkqhkiG9w0BAQsFADCBmDELMAkGA1UEBhMCVVMx
  EDAOBgNVBAgTB0FyaXpvbmExEzARBgNVBAcTClNjb3R0c2RhbGUxJTAjBgNVBAoT
  HFN0YXJmaWVsZCBUZWNobm9sb2dpZXMsIEluYy4xOzA5BgNVBAMTMlN0YXJmaWVs
  ZCBTZXJ2aWNlcyBSb290IENlcnRpZmljYXRlIEF1dG...
```

```
-----END CERTIFICATE-----"
```

Konfigurasi jaringan

Jika Anda menggunakan lingkungan Apache Kafka yang dikelola sendiri yang menggunakan konektivitas VPC pribadi, EventBridge harus memiliki akses ke sumber daya Amazon Virtual Private Cloud (Amazon VPC) yang terkait dengan broker Apache Kafka Anda. Untuk mengakses VPC cluster Apache Kafka Anda, EventBridge memerlukan akses internet keluar untuk subnet sumber Anda. Untuk subnet publik ini harus menjadi gateway NAT terkelola. Untuk subnet pribadi itu bisa menjadi gateway NAT, atau NAT Anda sendiri. Pastikan NAT memiliki alamat IP publik dan dapat terhubung ke internet.

Konfigurasikan grup keamanan Amazon VPC Anda dengan aturan berikut (minimal):

- Aturan masuk - Izinkan semua lalu lintas di port broker Apache Kafka (9092 untuk teks biasa, 9094 untuk TLS, 9096 untuk SASL, 9098 untuk IAM) untuk grup keamanan yang ditentukan untuk sumber Anda.
- Aturan keluar - Izinkan semua lalu lintas di port 443 untuk semua tujuan. Izinkan semua lalu lintas di port broker Apache Kafka (9092 untuk plaintext, 9094 untuk TLS, 9096 untuk SASL, 9098 untuk IAM) untuk grup keamanan yang ditentukan untuk sumber Anda.

Penskalaan otomatis konsumen dengan sumber Apache Kafka

Saat Anda awalnya membuat sumber Apache Kafka, EventBridge alokasikan satu konsumen untuk memproses semua partisi dalam topik Kafka. Setiap konsumen memiliki beberapa prosesor yang berjalan secara paralel untuk menangani peningkatan beban kerja. Selain itu, EventBridge secara otomatis meningkatkan atau menurunkan jumlah konsumen, berdasarkan beban kerja. Untuk mempertahankan pemesanan pesan di setiap partisi, jumlah maksimum konsumen adalah satu konsumen per partisi dalam topik.

Dalam interval satu menit, EventBridge mengevaluasi lag offset konsumen dari semua partisi dalam topik. Jika lag terlalu tinggi, partisi menerima pesan lebih cepat daripada yang EventBridge dapat memprosesnya. Jika perlu, EventBridge menambah atau menghapus konsumen dari topik. Proses penskalaan penambahan atau penghapusan konsumen terjadi dalam waktu tiga menit setelah evaluasi.

Jika target Anda kelebihan beban, EventBridge kurangi jumlah konsumen. Tindakan ini mengurangi beban kerja pada fungsi dengan mengurangi jumlah pesan yang dapat konsumen ambil dan kirim ke fungsi.

Layanan Antrian Sederhana Amazon sebagai sumber

Anda dapat menggunakan EventBridge Pipes untuk menerima catatan dari antrian Amazon SQS. Anda kemudian dapat secara opsional memfilter atau menyempurnakan catatan ini sebelum mengirimnya ke tujuan yang tersedia untuk diproses.

Anda dapat menggunakan pipa untuk memproses pesan dalam antrian Amazon Simple Queue Service (Amazon SQS). EventBridge Pipa mendukung antrian standar dan antrian first-in, first-out (FIFO). Dengan Amazon SQS, Anda dapat memindahkan tugas dari satu komponen aplikasi Anda dengan mengirimkannya ke antrean dan memprosesnya secara asinkron.

EventBridge polling antrian dan memanggil pipa Anda secara serempak dengan acara yang berisi pesan antrian. EventBridge membaca pesan dalam batch dan memanggil pipa Anda sekali untuk setiap batch. Ketika pipa Anda berhasil memproses batch, EventBridge menghapus pesannya dari antrian.

Secara default, EventBridge polling hingga 10 pesan dalam antrian Anda secara bersamaan dan mengirimkan batch itu ke pipa Anda. Untuk menghindari pemanggilan pipa dengan sejumlah kecil catatan, Anda dapat memberi tahu sumber acara untuk menyangga catatan hingga lima menit dengan mengonfigurasi jendela batch. Sebelum menjalankan pipa, EventBridge terus polling pesan dari antrian standar Amazon SQS hingga salah satu hal ini terjadi:

- Jendela batch kedaluwarsa.
- Kuota ukuran payload doa tercapai.
- Ukuran batch maksimum yang dikonfigurasi tercapai.

> **Note**
>
> Jika Anda menggunakan jendela batch dan antrean Amazon SQS berisi lalu lintas rendah, EventBridge mungkin menunggu hingga 20 detik sebelum menjalankan pipa Anda. Ini benar bahkan jika Anda mengatur jendela batch kurang dari 20 detik. Untuk antrean FIFO, rekaman berisi atribut tambahan yang terkait dengan deduplikasi dan pengurutan.

Saat EventBridge membaca batch, pesan tetap berada dalam antrian tetapi disembunyikan selama batas waktu visibilitas antrian. Jika pipa Anda berhasil memproses batch, EventBridge hapus pesan dari antrian. Secara default, jika pipa Anda mengalami kesalahan saat memproses batch, semua pesan dalam batch tersebut akan terlihat dalam antrian lagi. Untuk alasan ini, kode pipa Anda harus dapat memproses pesan yang sama beberapa kali tanpa efek samping yang tidak diinginkan. Anda dapat mengubah perilaku pemrosesan ulang ini dengan menyertakan kegagalan item batch dalam respons pipa Anda. Contoh berikut menunjukkan kejadian untuk batch yang berisi dua pesan.

Contoh peristiwa

Contoh peristiwa berikut menunjukkan informasi yang diterima oleh pipa. Anda dapat menggunakan acara ini untuk membuat dan memfilter pola acara Anda, atau untuk menentukan transformasi input. Tidak semua bidang dapat disaring. Untuk informasi selengkapnya tentang bidang mana yang dapat Anda filter, lihat???.

Antrian standar


```
[
  {
    "messageId": "059f36b4-87a3-44ab-83d2-661975830a7d",
    "receiptHandle": "AQEBwJnKyrHigUMZj6rYigCgxlaS3SLy0a...",
    "body": "Test message.",
    "attributes": {
      "ApproximateReceiveCount": "1",
      "SentTimestamp": "1545082649183",
      "SenderId": "AIDAIENQZJOLO23YVJ4VO",
      "ApproximateFirstReceiveTimestamp": "1545082649185"
    },
    "messageAttributes": {},
    "md5OfBody": "e4e68fb7bd0e697a0ae8f1bb342846b3",
    "eventSource": "aws:sqs",
    "eventSourceARN": "arn:aws:sqs:us-east-2:123456789012:my-queue",
    "awsRegion": "us-east-2"
  },
  {
    "messageId": "2e1424d4-f796-459a-8184-9c92662be6da",
    "receiptHandle": "AQEBzWwaftRI0KuVm4tP+/7q1rGgNqicHq...",
    "body": "Test message.",
    "attributes": {
      "ApproximateReceiveCount": "1",
      "SentTimestamp": "1545082650636",
      "SenderId": "AIDAIENQZJOLO23YVJ4VO",
```

```
      "ApproximateFirstReceiveTimestamp": "1545082650649"
    },
    "messageAttributes": {},
    "md5OfBody": "e4e68fb7bd0e697a0ae8f1bb342846b3",
    "eventSource": "aws:sqs",
    "eventSourceARN": "arn:aws:sqs:us-east-2:123456789012:my-queue",
    "awsRegion": "us-east-2"
  }
]
```

Antrian FIFO

```
[
  {
    "messageId": "11d6ee51-4cc7-4302-9e22-7cd8afdaadf5",
    "receiptHandle": "AQEBBX8nesZEXmkhsmZeyIE8iQAMig7qw...",
    "body": "Test message.",
    "attributes": {
      "ApproximateReceiveCount": "1",
      "SentTimestamp": "1573251510774",
      "SequenceNumber": "18849496460467696128",
      "MessageGroupId": "1",
      "SenderId": "AIDAIO23YVJENQZJOL4VO",
      "MessageDeduplicationId": "1",
      "ApproximateFirstReceiveTimestamp": "1573251510774"
    },
    "messageAttributes": {},
    "md5OfBody": "e4e68fb7bd0e697a0ae8f1bb342846b3",
    "eventSource": "aws:sqs",
    "eventSourceARN": "arn:aws:sqs:us-east-2:123456789012:fifo.fifo",
    "awsRegion": "us-east-2"
  }
]
```

Penskalaan dan pemrosesan

Untuk antrian standar, EventBridge gunakan polling panjang untuk polling antrian hingga menjadi aktif. Ketika pesan tersedia, EventBridge baca hingga lima batch dan kirimkan ke pipa Anda. Jika pesan masih tersedia, EventBridge tingkatkan jumlah proses yang membaca batch hingga 300 instance lagi per menit. Jumlah maksimum batch yang dapat diproses pipa secara bersamaan adalah 1.000.

Untuk antrian FIFO, EventBridge kirim pesan ke pipa Anda dalam urutan yang diterimanya. Saat Anda mengirimkan pesan ke antrean FIFO, Anda menentukan ID grup pesan. Amazon SQS memfasilitasi pengiriman pesan dalam grup yang sama ke EventBridge, secara berurutan. EventBridge mengurutkan pesan yang diterima ke dalam grup dan mengirim hanya satu batch pada satu waktu untuk grup. Jika pipa Anda mengembalikan kesalahan, pipa mencoba semua percobaan ulang pada pesan yang terpengaruh sebelum EventBridge menerima pesan tambahan dari grup yang sama.

Mengkonfigurasi antrian untuk digunakan dengan Pipes EventBridge

Buat antrian Amazon SQS untuk dijadikan sumber pipa Anda. Kemudian konfigurasikan antrian untuk memberikan waktu bagi pipa Anda untuk memproses setiap kumpulan peristiwa — dan untuk EventBridge mencoba lagi sebagai respons terhadap kesalahan pelambatan saat meningkatkan skala.

Untuk memungkinkan waktu pipa Anda memproses setiap batch rekaman, atur batas waktu visibilitas antrian sumber menjadi setidaknya enam kali runtime gabungan pengayaan pipa dan komponen target. Waktu tambahan memungkinkan untuk EventBridge mencoba lagi jika pipa Anda dibatasi saat memproses batch sebelumnya.

Jika pipa Anda gagal memproses pesan beberapa kali, Amazon SQS dapat mengirimkannya ke antrian surat mati. Ketika pipa Anda mengembalikan kesalahan, EventBridge simpan dalam antrian. Setelah batas waktu visibilitas terjadi, EventBridge terima pesan lagi. Untuk mengirim pesan ke antrean kedua setelah sejumlah penerimaan, konfigurasikan antrean surat gagal pada antrean sumber Anda.

> **Note**
>
> Pastikan Anda mengonfigurasi antrian huruf mati pada antrian sumber, bukan pada pipa. Antrian huruf mati yang Anda konfigurasikan pada pipa digunakan untuk antrian pemanggilan asinkron pipa, bukan untuk antrian sumber.

Jika pipa Anda mengembalikan kesalahan, atau tidak dapat dipanggil karena konkurensi maksimum, pemrosesan mungkin berhasil dengan upaya tambahan. Untuk memberikan pesan lebih banyak kesempatan untuk diproses sebelum mengirimnya ke antrian surat mati, setel kebijakan redrive antrian sumber ke setidaknya 5. **`maxReceiveCount`**

Melaporkan kegagalan item batch

Saat EventBridge mengkonsumsi dan memproses streaming data dari suatu sumber, secara default ia memeriksa ke nomor urutan tertinggi dari batch, tetapi hanya ketika batch berhasil total. Untuk menghindari pemrosesan ulang pesan yang berhasil diproses dalam kumpulan yang gagal, Anda dapat mengonfigurasi pengayaan atau target untuk mengembalikan objek yang menunjukkan pesan mana yang berhasil dan mana yang gagal. Ini disebut respon batch paral.

Untuk informasi selengkapnya, lihat ???.

Status berhasil dan gagal

Jika Anda mengembalikan salah satu dari yang berikut ini EventBridge , perlakukan batch sebagai kesuksesan total:

- Daftar `batchItemFailure` kosong
- Daftar `batchItemFailure` nol
- `EventResponse` kosong
- `EventResponse` nol

Jika Anda mengembalikan salah satu dari yang berikut ini, EventBridge memperlakukan batch sebagai kegagalan total:

- String `itemIdentifier` kosong
- `itemIdentifier` nol
- `itemIdentifier` dengan nama kunci yang buruk

EventBridge mencoba kembali kegagalan berdasarkan strategi coba lagi Anda.

Penyaringan EventBridge Pipa Amazon

Dengan EventBridge Pipes, Anda dapat memfilter peristiwa sumber tertentu dan memproses hanya sebagian dari mereka. Pemfilteran ini bekerja dengan cara yang sama seperti memfilter pada bus EventBridge acara atau pemetaan sumber peristiwa Lambda, dengan menggunakan pola peristiwa. Untuk informasi selengkapnya tentang pola acara, lihat???.

`FilterCriteria`Objek kriteria filter adalah struktur yang terdiri dari daftar filter (`Filters`). Setiap filter adalah struktur yang mendefinisikan pola penyaringan ()`Pattern`. A `Pattern` adalah

representasi string dari aturan filter JSON. Sebuah `FilterCriteria` objek terlihat seperti contoh berikut:

```
{
  "Filters": [
    {"Pattern": "{ \"Metadata1\": [ rule1 ], \"data\": { \"Data1\": [ rule2 ] }}"
    }
  ]
}
```

Untuk kejelasan tambahan, berikut adalah nilai filter yang `Pattern` diperluas di JSON biasa:

```
{
  "Metadata1": [ pattern1 ],
  "data": {"Data1": [ pattern2 ]}
}
```

Bagian utama untuk `FilterCriteria` objek adalah properti metadata dan properti data.

- Properti metadata adalah bidang objek acara. Dalam contoh, `FilterCriteria.Metadata1` mengacu pada properti metadata.
- Properti data adalah bidang badan acara. Dalam contoh, `FilterCriteria.Data1` mengacu pada properti data.

Misalnya, aliran Kinesis Anda berisi peristiwa seperti ini::

```
{
  "kinesisSchemaVersion": "1.0",
  "partitionKey": "1",
  "sequenceNumber": "49590338271490256608559692538361571095921575989136588898",
  "data": {"City": "Seattle",
    "State": "WA",
    "Temperature": "46",
    "Month": "December"
  },
  "approximateArrivalTimestamp": 1545084650.987
}
```

Ketika peristiwa mengalir melalui pipa Anda, itu akan terlihat seperti berikut dengan `data` bidang base64 yang dikodekan:

```
{
  "kinesisSchemaVersion": "1.0",
  "partitionKey": "1",
  "sequenceNumber": "49590338271490256608559692538361571095921575989136588898",
  "data": "SGVsbG8sIHRoaXMgaXMgYSB0ZXN0Lg==",
  "approximateArrivalTimestamp": 1545084650.987
  "eventSource": "aws:kinesis",
  "eventVersion": "1.0",
  "eventID":
 "shardId-000000000006:49590338271490256608559692538361571095921575989136588898",
  "eventName": "aws:kinesis:record",
  "invokeIdentityArn": "arn:aws:iam::123456789012:role/lambda-role",
  "awsRegion": "us-east-2",
  "eventSourceARN": "arn:aws:kinesis:us-east-2:123456789012:stream/lambda-stream"
},
```

Properti metadata pada peristiwa Kinesis adalah bidang apa pun di luar `data` objek, seperti atau. `partitionKey` `sequenceNumber`

Properti data dari peristiwa Kinesis adalah bidang di dalam `data` objek, seperti `City` atau. `Temperature`

Saat Anda memfilter agar sesuai dengan acara ini, Anda dapat menggunakan filter pada bidang yang diterjemahkan. Misalnya, untuk memfilter `partitionKey` dan `City` Anda akan menggunakan filter berikut:

```
{ "partitionKey": [ "1" ], "data": { "City": [ "Seattle" ] }
```

Saat Anda membuat filter acara, EventBridge Pipes dapat mengakses konten acara. Konten ini dapat diloloskan oleh JSON, seperti bidang Amazon SQS, atau yang dikodekan base64, seperti body bidang Kinesis. `data` Jika data Anda adalah JSON yang valid, template input atau jalur JSON Anda untuk parameter target dapat mereferensikan konten secara langsung. Misalnya, jika sumber peristiwa Kinesis adalah JSON yang valid, Anda dapat mereferensikan variabel menggunakan. < `$.data.someKey`>

Saat membuat pola peristiwa, Anda dapat memfilter berdasarkan bidang yang dikirim oleh API sumber, dan bukan pada bidang yang ditambahkan oleh operasi polling. Bidang berikut tidak dapat digunakan dalam pola acara:

- `awsRegion`

- `eventSource`
- `eventSourceARN`
- `eventVersion`
- `eventID`
- `eventName`
- `invokeIdentityArn`
- `eventSourceKey`

Bidang pesan dan data

Setiap sumber EventBridge Pipe berisi bidang yang berisi pesan inti atau data. Kami menyebutnya sebagai bidang pesan atau bidang data. Bidang ini istimewa karena mungkin JSON-escaped atau base64 dikodekan, tetapi ketika mereka adalah JSON yang valid, mereka dapat disaring dengan pola JSON seolah-olah tubuh tidak lolos. Isi bidang ini juga dapat digunakan dalam transformator input dengan mulus.

Memfilter pesan Amazon SQS dengan benar

Jika pesan Amazon SQS tidak memenuhi kriteria filter Anda, EventBridge secara otomatis menghapus pesan dari antrian. Anda tidak perlu menghapus pesan ini secara manual di Amazon SQS.

Untuk Amazon SQS, pesannya `body` bisa berupa string apa saja. Namun, ini bisa menjadi masalah jika Anda `FilterCriteria` berharap `body` berada dalam format JSON yang valid. Skenario sebaliknya juga benar —jika pesan `body` yang masuk dalam format JSON yang valid, tetapi kriteria filter Anda diharapkan menjadi string biasa, itu mengarah `body` ke perilaku yang tidak diinginkan.

Untuk menghindari masalah ini, pastikan bahwa format `body` dalam Anda `FilterCriteria` cocok dengan format yang diharapkan `body` dalam pesan yang Anda terima dari antrian Anda. Sebelum memfilter pesan Anda, EventBridge secara otomatis mengevaluasi format pesan masuk `body` dan pola filter Anda. `body` Jika ada ketidakcocokan, EventBridge jatuhkan pesan. Tabel berikut merangkum evaluasi ini:

Format pesan **body** masuk	**body**Format pola filter	Tindakan yang dihasilkan
Tali polos	Tali polos	EventBridge filter berdasarkan kriteria filter Anda.
Tali polos	Tidak ada pola filter untuk properti data	EventBridge filter (hanya pada properti metadata lainnya) berdasarkan kriteria filter Anda.
Tali polos	JSON yang valid	EventBridge menjatuhkan pesan.
JSON yang valid	Tali polos	EventBridge menjatuhkan pesan.
JSON yang valid	Tidak ada pola filter untuk properti data	EventBridge filter (hanya pada properti metadata lainnya) berdasarkan kriteria filter Anda.
JSON yang valid	JSON yang valid	EventBridge filter berdasarkan kriteria filter Anda.

Jika Anda tidak menyertakan `body` sebagai bagian dari Anda`FilterCriteria`, EventBridge lewati pemeriksaan ini.

Memfilter pesan Kinesis dan DynamoDB dengan benar

Setelah kriteria filter Anda memproses Kinesis atau rekaman DynamoDB, iterator stream akan melewati catatan ini. Jika catatan tidak memenuhi kriteria filter Anda, Anda tidak perlu menghapus rekaman secara manual dari sumber acara Anda. Setelah periode retensi, Kinesis dan DynamoDB secara otomatis menghapus catatan lama ini. Jika Anda ingin catatan dihapus lebih cepat, lihat Mengubah Periode Retensi Data.

Untuk memfilter peristiwa dengan benar dari sumber peristiwa aliran, bidang data dan kriteria filter Anda untuk bidang data harus dalam format JSON yang valid. (Untuk Kinesis, bidang data adalah. `data` Untuk DynamoDB, bidang datanya adalah.) `dynamodb` Jika salah satu bidang tidak dalam

format JSON yang valid, EventBridge jatuhkan pesan atau melempar pengecualian. Tabel berikut merangkum perilaku spesifik:

Format data masuk (**`data`**atau**`dynamodb`**)	Format pola filter untuk properti data	Tindakan yang dihasilkan
JSON yang valid	JSON yang valid	EventBridge filter berdasarkan kriteria filter Anda.
JSON yang valid	Tidak ada pola filter untuk properti data	EventBridge filter (hanya pada properti metadata lainnya) berdasarkan kriteria filter Anda.
JSON yang valid	Non-JSON	EventBridge melempar pengecualian pada saat pipa atau pembaruan. Pola filter untuk properti data harus dalam format JSON yang valid.
Non-JSON	JSON yang valid	EventBridge menjatuhkan rekor.
Non-JSON	Tidak ada pola filter untuk properti data	EventBridge filter (hanya pada properti metadata lainnya) berdasarkan kriteria filter Anda.
Non-JSON	Non-JSON	EventBridge melempar pengecualian pada saat pembuatan pipa atau pembaruan. Pola filter untuk properti data harus dalam format JSON yang valid.

Memfilter Amazon Managed Streaming dengan benar untuk Apache Kafka Kafka, Apache Kafka yang dikelola sendiri, dan pesan Amazon MQ

Untuk sumber Amazon MQ, bidang pesannya adalah. `data` Untuk sumber Apache Kafka (Amazon MSK dan Apache Kafka yang dikelola sendiri), ada dua bidang pesan: dan. `key` `value`

EventBridge menjatuhkan pesan yang tidak cocok dengan semua bidang yang disertakan dalam filter. Untuk Apache Kafka, EventBridge melakukan offset untuk pesan yang cocok dan tak tertandingi setelah berhasil menjalankan fungsi. Untuk Amazon MQ, EventBridge mengakui pesan yang cocok setelah berhasil menjalankan fungsi dan mengakui pesan yang tak tertandingi saat memfilternya.

Pesan Apache Kafka dan Amazon MQ harus berupa string yang dikodekan UTF-8, baik string biasa atau dalam format JSON. Itu karena EventBridge menerjemahkan array Apache Kafka dan Amazon MQ byte ke UTF-8 sebelum menerapkan kriteria filter. Jika pesan Anda menggunakan pengkodean lain, seperti UTF-16 atau ASCII, atau jika format pesan tidak cocok dengan format, EventBridge hanya memproses filter `FilterCriteria` metadata. Tabel berikut merangkum perilaku spesifik:

Format pesan masuk (**`data`**atau **`key`** dan**`value`**)	Format pola filter untuk properti pesan	Tindakan yang dihasilkan
Tali polos	Tali polos	EventBridge filter berdasarkan kriteria filter Anda.
Tali polos	Tidak ada pola filter untuk properti data	EventBridge filter (hanya pada properti metadata lainnya) berdasarkan kriteria filter Anda.
Tali polos	JSON yang valid	EventBridge filter (hanya pada properti metadata lainnya) berdasarkan kriteria filter Anda.
JSON yang valid	Tali polos	EventBridge filter (hanya pada properti metadata lainnya) berdasarkan kriteria filter Anda.

Format pesan masuk (**`data`**atau **`key`** dan**`value`**)	Format pola filter untuk properti pesan	Tindakan yang dihasilkan
JSON yang valid	Tidak ada pola filter untuk properti data	EventBridge filter (hanya pada properti metadata lainnya) berdasarkan kriteria filter Anda.
JSON yang valid	JSON yang valid	EventBridge filter berdasarkan kriteria filter Anda.
String yang tidak dikodekan UTF-8	JSON, string polos, atau tidak ada pola	EventBridge filter (hanya pada properti metadata lainnya) berdasarkan kriteria filter Anda.

Perbedaan antara Lambda ESM dan Pipa EventBridge

Saat memfilter acara, Lambda ESM EventBridge dan Pipes beroperasi secara umum dengan cara yang sama. Perbedaan utama adalah bahwa `eventSourceKey` bidang tidak ada dalam muatan ESM.

Pengayaan acara AmazonEventBridge Pipes

Dengan langkah pengayaanEventBridge Pipa, Anda dapat meningkatkan data dari sumber sebelum mengirimkannya ke target. Misalnya, Anda mungkin menerima peristiwa yang dibuat Tiket yang tidak menyertakan data tiket lengkap. Menggunakan enrichment, Anda dapat memiliki fungsi Lambda memanggil`get-ticket` API untuk detail tiket lengkap. Pipa kemudian dapat mengirim informasi itu ke target.

Anda dapat mengkonfigurasi pengayaan berikut saat menyiapkan pipa diEventBridge:

- Tujuan API
- Amazon API Gateway
- Fungsi Lambda
- Mesin status Step Functions

> **Note**
>
> EventBridgePipa hanya mendukung alur kerja Express sebagai pengayaan.

EventBridgememanggil pengayaan serempak karena harus menunggu respon dari pengayaan sebelum memohon target.

Dibatasi untuk ukuran maksimum 6MB.

Anda juga dapat mengubah data yang Anda terima dari sumber sebelum mengirimkannya untuk peningkatan. Untuk informasi selengkapnya, lihat ???.

Menyaring peristiwa menggunakan pengayaan

EventBridgePipa melewati respons pengayaan langsung ke target yang dikonfigurasi. Ini termasuk respons array untuk target yang mendukung batch. Untuk informasi selengkapnya tentang perilaku batch, lihat???. Anda juga dapat menggunakan pengayaan Anda sebagai filter dan melewati lebih sedikit peristiwa daripada yang diterima dari sumber. Jika Anda tidak ingin memanggil target, kembalikan respons kosong, seperti`""`,`{}`, atau`[]`.

> **Note**
>
> Jika Anda ingin memanggil target dengan payload kosong, kembalikan array dengan JSON kosong`[{}]`.

Memanggil pengayaan

EventBridgememanggil pengayaan secara sinkron (jenis pemanggilan diatur ke`REQUEST_RESPONSE`) karena harus menunggu respons dari pengayaan sebelum memanggil target.

> **Note**
>
> Untuk mesin status Step Functions,EventBridge hanya mendukung alur kerja Express sebagai pengayaan, karena dapat dipanggil secara sinkron.

Target Amazon EventBridge Pipes

Anda dapat mengirim data dalam pipa Anda ke target tertentu. Anda dapat mengonfigurasi target berikut saat menyiapkan pipa di EventBridge:

- Tujuan API
- API Gateway
- Antrian pekerjaan batch
- CloudWatch grup log
- Tugas ECS
- Bus acara di akun dan Wilayah yang sama
- Aliran pengiriman Firehose
- Templat penilaian Inspector
- Aliran kinesis
- Fungsi Lambda (SYNC atau ASYNC)
- Kueri API data klaster Redshift
- SageMaker Pipa
- Topik Amazon SNS (topik SNS FIFO tidak didukung)
- Antrean Amazon SQS
- Mesin status Step Functions
  - Alur kerja ekspres (SYNC atau ASYNC)
  - Alur kerja standar (ASYNC)

Parameter terget

Beberapa layanan target tidak mengirim payload acara ke target, sebaliknya, mereka memperlakukan acara sebagai pemicu untuk menjalankan API tertentu. EventBridge menggunakan `PipeTargetParameters`untuk menentukan informasi apa yang dikirim ke API itu. Sumber daya yang dimaksud meliputi:

- Tujuan API (Data yang dikirim ke tujuan API harus sesuai dengan struktur API. Anda harus menggunakan `InputTemplate`objek untuk memastikan data terstruktur dengan benar. Jika Anda ingin memasukkan muatan acara asli, rujuk di `InputTemplate`.)

- API Gateway (Data yang dikirim ke API Gateway harus sesuai dengan struktur API. Anda harus menggunakan `InputTemplate`objek untuk memastikan data terstruktur dengan benar. Jika Anda ingin memasukkan muatan acara asli, rujuk di `InputTemplate`.)
- `PipeTargetRedshiftDataParameters`(Cluster API Data Amazon Redshift)
- `PipeTargetSageMakerPipelineParameters`(Pipa Pembuatan Model SageMaker Runtime Amazon)
- `PipeTargetBatchJobParameters` (AWS Batch)

> **Note**
>
> EventBridge tidak mendukung semua sintaks JSON Path dan mengevaluasinya saat runtime. Sintaks yang didukung meliputi:
>
> - notasi titik (misalnya,`$.detail`)
> - garis putus-putus
> - menggarisbawahi
> - karakter alfanumerik
> - indeks array
> - wildcard (*)

Parameter jalur dinamis

EventBridge Parameter target pipa mendukung sintaks jalur JSON dinamis opsional. Anda dapat menggunakan sintaks ini untuk menentukan jalur JSON alih-alih nilai statis (misalnya`$.detail.state`). Seluruh nilai harus berupa jalur JSON, tidak hanya sebagian saja. Misalnya, `RedshiftParameters.Sql` bisa `$.detail.state` tetapi tidak bisa`"SELECT * FROM $.detail.state"`. Jalur ini diganti secara dinamis saat runtime dengan data dari payload acara itu sendiri di jalur yang ditentukan. Parameter jalur dinamis tidak dapat mereferensikan nilai baru atau yang diubah yang dihasilkan dari transformasi input. Sintaks yang didukung untuk jalur JSON parameter dinamis sama seperti saat mengubah input. Untuk informasi selengkapnya, lihat ???.

Sintaks dinamis dapat digunakan pada semua string, bidang non-enum dari semua pengayaan EventBridge Pipa dan parameter target kecuali:

- `PipeTargetCloudWatchLogsParameters.LogStreamName`

- `PipeTargetEventBridgeEventBusParameters.EndpointId`
- `PipeEnrichmentHttpParameters.HeaderParameters`
- `PipeTargetHttpParameters.HeaderParameters`

Misalnya, untuk menyetel target Kinesis pipa ke kunci kustom dari peristiwa sumber Anda, atur. `PartitionKey` KinesisTargetParameter PartitionKeyke:

- `"$.data.`*`someKey`*`"`untuk sumber Kinesis
- `"$.body.`*`someKey`*`"`untuk sumber Amazon SQS

Kemudian, jika payload acara adalah string JSON yang valid, seperti`{"`*`someKey`*`":"`*`someValue`*`"}`, EventBridge mengekstrak nilai dari jalur JSON dan menggunakannya sebagai parameter target. Dalam contoh ini, EventBridge akan mengatur Kinesis `PartitionKey` ke "*`SomeValue`*”.

Izin

Untuk melakukan panggilan API pada sumber daya yang Anda miliki, EventBridge Pipes memerlukan izin yang sesuai. EventBridge PIPEs menggunakan peran IAM yang Anda tentukan pada pipa untuk pengayaan dan panggilan target menggunakan prinsipal IAM. `pipes.amazonaws.com`

Memanggil target

EventBridge memiliki cara berikut untuk memanggil target:

- Synchronously (tipe pemanggilan diatur ke`REQUEST_RESPONSE`) — EventBridge menunggu respons dari target sebelum melanjutkan.
- Secara asinkron (tipe pemanggilan disetel ke`FIRE_AND_FORGET`) — EventBridge tidak menunggu respons sebelum melanjutkan.

Secara default, untuk pipa dengan sumber yang dipesan, EventBridge memanggil target secara serempak karena respons dari target diperlukan sebelum melanjutkan ke acara berikutnya.

Jika sumber tidak menerapkan urutan, seperti antrian EventBridge Amazon SQS standar, dapat memanggil target yang didukung secara sinkron atau asinkron.

Dengan fungsi Lambda dan mesin status Step Functions, Anda dapat mengonfigurasi jenis pemanggilan.

> **Note**
>
> Untuk mesin status Step Functions, alur kerja Standar harus dipanggil secara asinkron.

EventBridge Pipa target spesifik

Antrean tugas AWS Batch

Semua AWS Batch `submitJob` parameter dikonfigurasi secara eksplisit dengan`BatchParameters`, dan seperti semua parameter Pipe, parameter ini dapat dinamis menggunakan jalur JSON ke muatan acara masuk Anda.

CloudWatch Grup log

Apakah Anda menggunakan transformator input atau tidak, muatan peristiwa digunakan sebagai pesan log. Anda dapat mengatur `Timestamp` (atau eksplisit `LogStreamName` tujuan Anda) melalui `CloudWatchLogsParameters` in`PipeTarget`. Seperti semua parameter pipa, parameter ini dapat dinamis saat menggunakan jalur JSON ke muatan acara masuk Anda.

Tugas Amazon ECS

Semua `runTask` parameter Amazon ECS dikonfigurasi secara eksplisit melalui. `EcsParameters` Seperti semua parameter pipa, parameter ini dapat dinamis saat menggunakan jalur JSON ke muatan acara masuk Anda.

Fungsi Lambda dan alur kerja Step Functions

Lambda dan Step Functions tidak memiliki API batch. Untuk memproses batch peristiwa dari sumber pipa, batch dikonversi ke array JSON dan diteruskan sebagai input ke target Lambda atau Step Functions. Untuk informasi selengkapnya, lihat ???.

Pengelompokan dan konkurensi EventBridge Pipa Amazon

Perilaku batching

EventBridge Pipa mendukung batching dari sumber dan ke target yang mendukungnya. Selain itu, batching hingga pengayaan didukung untuk dan. AWS Lambda AWS Step Functions Karena layanan yang berbeda mendukung tingkat batching yang berbeda, Anda tidak dapat mengonfigurasi pipa

dengan ukuran batch yang lebih besar daripada dukungan target. Misalnya, sumber aliran Amazon Kinesis mendukung ukuran batch maksimum 10.000 rekaman, tetapi Layanan Antrian Sederhana Amazon mendukung maksimum 10 pesan per batch sebagai target. Oleh karena itu, pipa dari aliran Kinesis ke antrian Amazon SQS dapat memiliki ukuran batch maksimum yang dikonfigurasi pada sumber 10.

Jika Anda mengonfigurasi pipa dengan pengayaan atau target yang tidak mendukung batching, Anda tidak akan dapat mengaktifkan batching pada sumbernya.

Saat batching diaktifkan pada sumber, array catatan JSON dilewatkan melalui pipa dan kemudian dipetakan ke API batch dari pengayaan atau target yang didukung. Transformator input diterapkan secara terpisah pada setiap catatan JSON individu dalam array, bukan array secara keseluruhan. Untuk contoh array ini, lihat ??? dan pilih sumber tertentu. Pipa akan menggunakan API batch untuk pengayaan atau target yang didukung meskipun ukuran batch adalah 1. Jika pengayaan atau target tidak memiliki API batch tetapi menerima muatan JSON penuh, seperti Lambda dan Step Functions, seluruh array JSON dikirim dalam satu permintaan. Permintaan akan dikirim sebagai array JSON bahkan jika ukuran batch adalah 1.

Jika pipa dikonfigurasi untuk batching di sumber, dan target mendukung batching, Anda dapat mengembalikan array item JSON dari pengayaan Anda. Array ini dapat mencakup array yang lebih pendek atau lebih panjang dari sumber aslinya. Namun, jika array lebih besar dari ukuran batch yang didukung oleh target, pipa tidak akan memanggil target.

Target batchable yang didukung

Target	Ukuran batch maksimum
CloudWatch Log	10.000
EventBridge bus acara	10
Aliran Firehose	500
Aliran kinesis	500
Fungsi Lambda	pelanggan didefinisikan
Mesin status Step Functions	pelanggan didefinisikan
Topik Amazon SNS	10

Target	Ukuran batch maksimum
Antrean Amazon SQS	10

Pengayaan dan target berikut menerima muatan peristiwa batch penuh untuk diproses dan dibatasi oleh ukuran muatan total acara, bukan ukuran batch:

- Mesin status Step Functions (262144 karakter)
- Fungsi Lambda (6MB)

Kegagalan batch sebagian

Untuk Amazon SQS dan sumber aliran, seperti Kinesis dan DynamoDB, Pipes mendukung penanganan kegagalan batch sebagian dari EventBridge kegagalan target. Jika target mendukung batching dan hanya sebagian dari batch yang berhasil, EventBridge secara otomatis mencoba kembali batching sisa muatan. Untuk konten yang paling up-to-date diperkaya, percobaan ulang ini terjadi melalui seluruh pipa, termasuk memanggil kembali pengayaan yang dikonfigurasi.

Penanganan kegagalan batch sebagian dari pengayaan tidak didukung.

Untuk target Lambda dan Step Functions, Anda juga dapat menentukan kegagalan sebagian dengan mengembalikan payload dengan struktur yang ditentukan dari target. Ini menunjukkan peristiwa yang perlu dicoba lagi.

Contoh struktur payload kegagalan sebagian

```
{
  "batchItemFailures": [
    {
      "itemIdentifier": "id2"
    },
    {
      "itemIdentifier": "id4"
    }
]
```

Dalam contoh, `itemIdentifier` mencocokkan ID peristiwa yang ditangani oleh target Anda dari sumber aslinya. Untuk Amazon SQS, ini adalah. `messageId` Untuk Kinesis dan DynamoDB, ini adalah. `eventID` Agar EventBridge Pipes dapat menangani kegagalan batch sebagian dari target

secara memadai, bidang ini harus disertakan dalam muatan array apa pun yang dikembalikan oleh pengayaan.

Perilaku throughput dan konkurensi

Setiap peristiwa atau kumpulan peristiwa yang diterima oleh pipa yang melakukan perjalanan ke pengayaan atau target dianggap sebagai eksekusi pipa. Pipa dalam `STARTED` keadaan terus melakukan polling untuk peristiwa dari sumber, meningkatkan dan menurunkan tergantung pada backlog yang tersedia dan pengaturan batching yang dikonfigurasi.

Untuk kuota eksekusi pipa bersamaan, dan jumlah pipa per akun dan Wilayah, lihat. ???

Secara default, satu pipa akan menskalakan ke eksekusi bersamaan maksimum berikut, tergantung pada sumbernya:

- DynamoDB — Eksekusi bersamaan dapat naik setinggi yang dikonfigurasi `ParallelizationFactor` pada pipa dikalikan dengan jumlah pecahan di sungai.
- Apache Kafka — Eksekusi bersamaan dapat naik setinggi jumlah partisi pada topik, hingga 1000.
- Kinesis — Eksekusi bersamaan dapat naik setinggi yang `ParallelizationFactor` dikonfigurasi pada pipa dikalikan dengan jumlah pecahan di sungai.
- Amazon MQ - 5
- Amazon SQS - 1250

Jika Anda memiliki persyaratan untuk throughput polling maksimum yang lebih tinggi atau batas konkurensi, hubungi dukungan.

> **Note**
>
> Batas eksekusi dianggap sebagai batasan keselamatan upaya terbaik. Meskipun polling tidak dibatasi di bawah nilai-nilai ini, pipa atau akun mungkin meledak lebih tinggi dari nilai rekomendasi ini.

Eksekusi pipa dibatasi maksimal 5 menit termasuk pengayaan dan pemrosesan target. Batas ini saat ini tidak dapat ditingkatkan.

Pipa dengan sumber yang diurutkan secara ketat, seperti antrian Amazon SQS FIFO, Kinesis dan DynamoDB Streams, atau topik Apache Kafka) selanjutnya dibatasi secara konkurensi oleh

konfigurasi sumber, seperti jumlah ID grup pesan untuk antrian FIFO atau jumlah pecahan untuk antrian Kinesis. Karena pemesanan dijamin ketat dalam batasan ini, pipa dengan sumber yang dipesan tidak dapat melebihi batas konkurensi tersebut.

AmazonEventBridgeTransformasi masukan pipa

AmazonEventBridgePipa mendukung transformator input opsional saat meneruskan data ke pengayaan dan target. Anda dapat menggunakan transformator Input untuk membentuk kembali payload input peristiwa JSON untuk melayani kebutuhan pengayaan atau layanan target. Untuk tujuan Amazon API Gateway dan API, inilah cara Anda membentuk peristiwa input ke model RESTful API Anda. Transformator input dimodelkan sebagai`InputTemplate`parameter. Mereka dapat berupa teks gratis, jalur JSON ke payload acara, atau objek JSON yang menyertakan jalur JSON sebaris ke payload acara. Untuk pengayaan, muatan acara berasal dari sumbernya. Untuk target, muatan acara adalah apa yang dikembalikan dari pengayaan, jika ada yang dikonfigurasi pada pipa. Selain data khusus layanan dalam muatan acara, Anda dapat menggunakanvariabel cadangandalam dirimu`InputTemplate`untuk referensi data untuk pipa Anda.

Untuk mengakses item dalam array, gunakan notasi braket persegi.

> **Note**
>
> EventBridgetidak mendukung semua sintaks JSON Path dan mengevaluasinya saat runtime. Sintaks yang didukung meliputi:
>
> - notasi titik (misalnya,`$.detail`)
> - tanda hubung
> - menggarisbawahi
> - karakter alfanumerik
> - indeks array
> - wildcard (*)

Berikut ini adalah contoh`InputTemplate`parameter yang mereferensikan payload peristiwa Amazon SQS:

String statis

```
InputTemplate: "Hello, sender"
```

Jalur JSON

```
InputTemplate: <$.attributes.SenderId>
```

String dinamis

```
InputTemplate: "Hello, <$.attributes.SenderId>"
```

JSON statis

```
InputTemplate: >
{
  "key1": "value1",
  "key2": "value2",
  "key3": "value3",
}
```

JSON Dinamis

```
InputTemplate: >
{
  "key1": "value1"
  "key2": <$.body.key>,
  "d": <aws.pipes.event.ingestion-time>
}
```

Menggunakan notasi braket persegi untuk mengakses item dalam array:

```
InputTemplate: >
{
  "key1": "value1"
  "key2": <$.body.Records[3]>,
  "d": <aws.pipes.event.ingestion-time>
}
```

> **Note**
>
> EventBridgemenggantikan transformator input saat runtime untuk memastikan output JSON yang valid. Karena itu, letakkan tanda kutip di sekitar variabel yang merujuk ke parameter jalur JSON, tetapi jangan beri tanda kutip di sekitar variabel yang merujuk ke objek atau array JSON.

Variabel yang dicadangkan

Template masukan dapat menggunakan variabel cadangan berikut:

- `<aws.pipes.pipe-arn>`— Nama Sumber Daya Amazon (ARN) dari pipa.
- `<aws.pipes.pipe-name>`— Nama pipa.
- `<aws.pipes.source-arn>`— ARN dari sumber acara pipa.
- `<aws.pipes.enrichment-arn>`— ARN pengayaan pipa.
- `<aws.pipes.target-arn>`— ARN dari target pipa.
- `<aws.pipes.event.ingestion-time>`— Waktu di mana acara diterima oleh transformator input. Ini adalah stempel waktu ISO 8601. Kali ini berbeda untuk trafo input pengayaan dan trafo input target, tergantung pada kapan pengayaan selesai memproses acara.
- `<aws.pipes.event>`— Acara yang diterima oleh transformator input.

  Untuk transformator input pengayaan, ini adalah acara dari sumbernya. Ini berisi muatan asli dari sumbernya, ditambah metadata khusus layanan tambahan. Lihat topiknya di???untuk contoh khusus layanan.

  Untuk transformator input target, ini adalah peristiwa yang dikembalikan oleh pengayaan, jika dikonfigurasi, tanpa metadata tambahan. Dengan demikian, muatan yang dikembalikan pengayaan mungkin non-JSON. Jika tidak ada pengayaan yang dikonfigurasi pada pipa, ini adalah peristiwa dari sumber dengan metadata.
- `<aws.pipes.event.json>`- Sama seperti`aws.pipes.event`, tetapi variabel hanya memiliki nilai jika muatan asli, baik dari sumber atau dikembalikan oleh pengayaan, adalah JSON. Jika pipa memiliki bidang yang dikodekan, seperti Amazon SQS`body`bidang atau Kinesis`data`, bidang-bidang tersebut diterjemahkan dan diubah menjadi JSON yang valid. Karena tidak lolos, variabel hanya dapat digunakan sebagai nilai untuk bidang JSON. Untuk informasi selengkapnya, lihat ???.

Contoh transformasi masukan

Berikut ini adalah contoh acara Amazon EC2 yang dapat kita gunakan sebagaicontoh acara.

```
{
  "version": "0",
  "id": "7bf73129-1428-4cd3-a780-95db273d1602",
  "detail-type": "EC2 Instance State-change Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2015-11-11T21:29:54Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2:us-east-1:123456789012:instance/i-abcd1111"
  ],
  "detail": {
    "instance-id": "i-0123456789",
    "state": "RUNNING"
  }
}
```

Mari kita gunakan JSON berikut sebagaiTransformator.

```
{
  "instance" : <$.detail.instance-id>,
  "state": <$.detail.state>,
  "pipeArn" : <aws.pipes.pipe-arn>,
  "pipeName" : <aws.pipes.pipe-name>,
  "originalEvent" : <aws.pipes.event.json>
}
```

Berikut ini akan menjadi hasilKeluaran:

```
{
  "instance" : "i-0123456789",
  "state": "RUNNING",
  "pipeArn" : "arn:aws:pipe:us-east-1:123456789012:pipe/example",
  "pipeName" : "example",
  "originalEvent" : {
    ... // commented for brevity
```

```
  }
}
```

Penguraian data tubuh implisit

Bidang berikut dalam payload yang masuk mungkin di-escaped JSON, seperti Amazon SQS`body`objek, atau dikodekan base64, seperti Kinesis`data`objek. Untuk keduanyamenyaringdan transformasi masukan,EventBridgemengubah bidang ini menjadi JSON yang valid sehingga sub-nilai dapat direferensikan secara langsung. Sebagai contoh,`<$.data.someKey>`untuk Kinesis.

Agar target menerima muatan asli tanpa metadata tambahan, gunakan transformator input dengan data tubuh ini, khusus untuk sumbernya. Sebagai contoh,`<$.body>`untuk Amazon SQS, atau`<$.data>`untuk Kinesis. Jika payload asli adalah string JSON yang valid (misalnya`{"key": "value"}`), kemudian penggunaan transformator input dengan data tubuh spesifik sumber akan menghasilkan tanda kutip dalam muatan sumber asli dihapus. Sebagai contoh,`{"key": "value"}`akan menjadi`"{key: value}"`ketika dikirim ke target. Jika target Anda memerlukan muatan JSON yang valid (misalnya,EventBridgeLambda atau Step Functions), ini akan menyebabkan kegagalan pengiriman. Agar target menerima data sumber asli tanpa menghasilkan JSON yang tidak valid, bungkus transformator input data badan sumber di JSON. Sebagai contoh, `{"data": <$.data>}`.

Penguraian tubuh implisit juga dapat digunakan untuk mengisi nilai secara dinamis untuk sebagian besar target pipa atau parameter pengayaan. Untuk informasi selengkapnya, lihat ???

> **Note**
>
> Jika payload asli adalah JSON yang valid, bidang ini akan berisi JSON yang tidak di-base64 yang tidak di-encode. Namun, jika payload tidak valid JSON,EventBridgebase64-encode untuk bidang yang tercantum di bawah ini, dengan pengecualian Amazon SQS.

- MQ Aktif—`data`
- Kinesis—`data`
- MSK Amazon—`key`dan`value`
- Kelinci MQ—`data`
- Apache Kafka yang dikelola sendiri;—`key`dan`value`
- Amazon SQS—body

Masalah umum dengan mengubah input

Ini adalah beberapa masalah umum saat mengubah inputEventBridgepipa:

- Untuk String, tanda kutip diperlukan.
- Tidak ada validasi saat membuat jalur JSON untuk templat Anda.
- Jika Anda menentukan variabel untuk mencocokkan jalur JSON yang tidak ada dalam peristiwa, variabel tersebut tidak dibuat dan tidak akan muncul dalam keluaran.
- Properti JSON seperti`aws.pipes.event.json`hanya dapat digunakan sebagai nilai bidang JSON, bukan sebaris di string lain.
- EventBridgetidak lolos dari nilai yang diekstraksi olehJalur Masukan, saat mengisiTemplate Masukanuntuk target.
- Jika jalur JSON mereferensikan objek atau array JSON, tetapi variabel direferensikan dalam string,EventBridgemenghapus tanda kutip internal untuk memastikan string yang valid. Misalnya, “Body is <$.body>” akan menghasilkanEventBridgemenghapus tanda kutip dari objek.

  Oleh karena itu, jika Anda ingin menampilkan objek JSON berdasarkan variabel jalur JSON tunggal, Anda harus menempatkannya sebagai kunci. Dalam contoh ini,`{"body": <$.body>}`.
- Kutipan tidak diperlukan untuk variabel yang mewakili string. Mereka diizinkan, tetapiEventBridgePipa secara otomatis menambahkan tanda kutip ke nilai variabel string selama transformasi, untuk memastikan output transformasi adalah JSON yang valid. EventBridgePipa tidak menambahkan tanda kutip ke variabel yang mewakili objek atau array JSON. Jangan menambahkan tanda kutip untuk variabel yang mewakili objek JSON atau array.

  Misalnya, template input berikut mencakup variabel yang mewakili string dan objek JSON:

```
{
  "pipeArn" : <aws.pipes.pipe-arn>,
  "pipeName" : <aws.pipes.pipe-name>,
  "originalEvent" : <aws.pipes.event.json>
}
```

  Menghasilkan JSON yang valid dengan kutipan yang tepat:

```
{
  "pipeArn" : "arn:aws:events:us-east-2:123456789012:pipe/example",
  "pipeName" : "example",
```

```
  "originalEvent" : {
    ... // commented for brevity
  }
}
```

- Untuk pengayaan atau target Lambda atau Step Functions, batch dikirim ke target sebagai array JSON, meskipun ukuran batch adalah 1. Namun, transformator input masih akan diterapkan ke catatan individu di JSON Array, bukan array secara keseluruhan. Untuk informasi selengkapnya, lihat ???.

Log EventBridge Pipa Amazon

EventBridge Pencatatan pipa memungkinkan Anda meminta EventBridge Pipa mengirim catatan yang merinci kinerja pipa ke AWS layanan yang didukung. Gunakan log untuk mendapatkan wawasan tentang kinerja eksekusi pipa Anda, dan untuk membantu pemecahan masalah dan debugging.

Anda dapat memilih AWS layanan berikut sebagai tujuan log tempat EventBridge Pipes mengirimkan catatan:

- CloudWatch Log

  EventBridge mengirimkan catatan log ke grup log CloudWatch Log yang ditentukan.

  Gunakan CloudWatch Log untuk memusatkan log dari semua sistem, aplikasi, dan AWS layanan yang Anda gunakan, dalam satu layanan yang sangat skalabel. Untuk informasi selengkapnya, lihat Bekerja dengan grup log dan aliran log di Panduan Pengguna CloudWatch Log Amazon.
- Log aliran Firehose

  EventBridge mengirimkan catatan log ke aliran pengiriman Firehose.

  Amazon Data Firehose adalah layanan yang dikelola sepenuhnya untuk mengirimkan data streaming real-time ke tujuan seperti AWS layanan tertentu, serta titik akhir HTTP kustom atau titik akhir HTTP yang dimiliki oleh penyedia layanan pihak ketiga yang didukung. Untuk informasi selengkapnya, lihat Membuat aliran pengiriman Amazon Data Firehose di Panduan Pengguna Amazon Data Firehose.
- Log Amazon S3

  EventBridge mengirimkan catatan log sebagai objek Amazon S3 ke bucket yang ditentukan.

Amazon S3 adalah layanan penyimpanan objek yang menawarkan skalabilitas, ketersediaan data, keamanan, dan kinerja terdepan di industri. Untuk informasi selengkapnya, lihat Mengunggah, mengunduh, dan bekerja dengan objek di Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Cara kerja pencatatan Amazon EventBridge Pipes

Eksekusi pipa adalah peristiwa atau kumpulan peristiwa yang diterima oleh pipa yang melakukan perjalanan ke pengayaan dan/atau target. Jika diaktifkan, EventBridge buat catatan log untuk setiap langkah eksekusi yang dilakukannya saat batch acara diproses. Informasi yang terkandung dalam catatan berlaku untuk batch acara, baik itu acara tunggal atau hingga 10.000 acara.

Anda dapat mengonfigurasi ukuran batch acara pada sumber pipa dan target. Untuk informasi selengkapnya, lihat ???.

Data rekaman yang dikirim ke setiap tujuan log adalah sama.

Jika tujuan Amazon CloudWatch Logs dikonfigurasi, catatan log yang dikirimkan ke semua tujuan memiliki batas 256kb. Bidang akan dipotong seperlunya.

Anda dapat menyesuaikan catatan yang EventBridge dikirim ke tujuan log yang dipilih dengan cara berikut:

- Anda dapat menentukan tingkat log, yang menentukan langkah-langkah eksekusi yang EventBridge mengirimkan catatan ke tujuan log yang dipilih. Untuk informasi selengkapnya, lihat ???.
- Anda dapat menentukan apakah EventBridge Pipes menyertakan data eksekusi dalam catatan untuk langkah-langkah eksekusi yang relevan. Data ini meliputi:
  - Muatan batch acara
  - Permintaan dikirim ke AWS pengayaan atau layanan target
  - Respons yang dikembalikan oleh AWS pengayaan atau layanan target

  Untuk informasi selengkapnya, lihat ???.

Menentukan tingkat log EventBridge Pipa

Anda dapat menentukan jenis langkah eksekusi yang EventBridge mengirimkan catatan ke tujuan log yang dipilih.

Pilih dari tingkat detail berikut untuk disertakan dalam catatan log. Tingkat log berlaku untuk semua tujuan log yang ditentukan untuk pipa. Setiap level log mencakup langkah-langkah eksekusi level log sebelumnya.

- OFF - EventBridge tidak mengirim catatan apa pun ke tujuan log tertentu. Ini adalah pengaturan default.
- ERROR — EventBridge mengirimkan catatan apa pun yang terkait dengan kesalahan yang dihasilkan selama eksekusi pipa ke tujuan log yang ditentukan.
- INFO — EventBridge mengirimkan catatan apa pun yang terkait dengan kesalahan, serta memilih langkah lain yang dilakukan selama eksekusi pipa ke tujuan log yang ditentukan.
- TRACE - EventBridge mengirimkan catatan apa pun yang dihasilkan selama langkah apa pun dalam eksekusi pipa ke tujuan log yang ditentukan.

Di EventBridge konsol, CloudWatch log dipilih sebagai tujuan log secara default, seperti tingkat `ERROR` log. Jadi, secara default, EventBridge Pipes membuat grup CloudWatch log baru yang mengirimkan catatan log yang berisi `ERROR` tingkat detail. Tidak ada default yang dipilih saat Anda mengonfigurasi log secara terprogram.

Tabel berikut mencantumkan langkah-langkah eksekusi yang disertakan dalam setiap tingkat log.

Langkah	JEJAK	INFO	ERROR	MATI
Eksekusi Gagal	x	x	x	
Eksekusi Sebagian Gagal	x	x	x	
Eksekusi Dimulai	x	x		
Eksekusi Berhasil	x	x		
Eksekusi Dibatasi	x	x	x	
Batas Waktu Eksekusi	x	x	x	

Langkah	JEJAK	INFO	ERROR	MATI
Doa Pengayaan Gagal	x	x	x	
Doa Pengayaan Dilewati	x	x		
Doa Pengayaan Dimulai	x			
Doa Pengayaan Berhasil	x			
Tahap Pengayaan Dimasukkan	x	x		
Tahap Pengayaan Gagal	x	x	x	
Tahap Pengayaan Berhasil	x	x		
Transformasi Pengayaan Gagal	x	x	x	
Transformasi Pengayaan Dimulai	x			
Transformasi Pengayaan Berhasil	x			
Doa Target Gagal	x	x	x	
Doa Target Sebagian Gagal	x	x	x	
Doa Target Dilewati	x			
Doa Target Dimulai	x			
Doa Target Berhasil	x			
Tahap Target Dimasukkan	x	x		
Tahap Target Gagal	x	x	x	
Tahap Target Sebagian Gagal	x	x	x	

Langkah	JEJAK	INFO	ERROR	MATI
Tahap Target Dilewati	x			
Tahap Target Berhasil	x	x		
Transformasi Target Gagal	x	x	x	
Transformasi Target Dimulai	x			
Transformasi Target Berhasil	x			

Termasuk data eksekusi di log EventBridge Pipes

Anda dapat menentukan EventBridge untuk menyertakan data eksekusi dalam catatan yang dihasilkannya. Data eksekusi mencakup bidang yang mewakili muatan batch peristiwa, serta permintaan yang dikirim ke dan respons dari pengayaan dan target.

Data eksekusi berguna untuk pemecahan masalah dan debugging. `payload`Bidang berisi konten aktual dari setiap peristiwa yang disertakan dalam batch, memungkinkan Anda untuk menghubungkan peristiwa individual dengan eksekusi pipa tertentu.

Jika Anda memilih untuk menyertakan data eksekusi, itu termasuk untuk semua tujuan log yang ditentukan untuk pipa.

> **Important**
>
> Bidang ini mungkin berisi informasi sensitif. EventBridge tidak berusaha untuk menyunting konten bidang ini selama pencatatan.

Saat menyertakan data eksekusi, EventBridge tambahkan bidang berikut ke catatan yang relevan:

- **`payload`**

  Merupakan isi dari batch acara yang sedang diproses oleh pipa.

  EventBridge termasuk `payload` bidang dalam catatan yang dihasilkan pada langkah-langkah di mana konten batch acara mungkin telah diperbarui. Ini termasuk langkah-langkah berikut:

  - EXECUTION_STARTED
  - ENRICHMENT_TRANSFORMATION_SUCCEEDED
  - ENRICHMENT_STAGE_SUCCEEDED
  - TARGET_TRANSFORMATION_SUCCEEDED
  - TARGET_STAGE_SUCCEEDED
- **awsRequest**

  Merupakan permintaan yang dikirim ke pengayaan atau target sebagai string JSON. Untuk permintaan yang dikirim ke tujuan API, ini mewakili permintaan HTTP yang dikirim ke titik akhir tersebut.

  EventBridge mencakup `awsRequest` bidang dalam catatan yang dihasilkan pada langkah akhir pengayaan dan penargetan; yaitu, setelah EventBridge mengeksekusi atau mencoba untuk mengeksekusi permintaan terhadap pengayaan atau layanan target yang ditentukan. Ini termasuk langkah-langkah berikut:

  - ENRICHMENT_INVOCATION_FAILED
  - ENRICHMENT_INVOCATION_SUCCEEDED
  - TARGET_INVOCATION_FAILED
  - TARGET_INVOCATION_PARTIALLY_FAILED
  - TARGET_INVOCATION_SUCCEEDED
- **awsResponse**

  Merupakan respons yang dikembalikan oleh pengayaan atau target, dalam format JSON. Untuk permintaan yang dikirim ke tujuan API, ini mewakili respons HTTP yang dikembalikan dari titik akhir tersebut.

  Seperti halnya`awsRequest`, EventBridge termasuk `awsResponse` bidang dalam catatan yang dihasilkan pada langkah akhir pengayaan dan penargetan; yaitu, setelah EventBridge mengeksekusi atau mencoba untuk mengeksekusi permintaan terhadap pengayaan atau layanan target yang ditentukan dan menerima tanggapan. Ini termasuk langkah-langkah berikut:

  - ENRICHMENT_INVOCATION_FAILED
  - ENRICHMENT_INVOCATION_SUCCEEDED
  - TARGET_INVOCATION_FAILED
  - TARGET_INVOCATION_PARTIALLY_FAILED

- `TARGET_INVOCATION_SUCCEEDED`

Untuk diskusi tentang langkah-langkah eksekusi pipa, lihat???.

Mempotong data eksekusi dalam catatan log EventBridge Pipes

Jika Anda memilih untuk EventBridge menyertakan data eksekusi dalam catatan log pipa, ada kemungkinan bahwa catatan dapat melebihi batas ukuran 256 KB. Untuk mencegah hal ini, EventBridge secara otomatis memotong bidang data eksekusi, dalam urutan berikut. EventBridge memotong setiap bidang seluruhnya sebelum melanjutkan untuk memotong bidang berikutnya. EventBridge memotong data bidang hanya dengan menghapus karakter dari akhir string data; tidak ada upaya yang dilakukan untuk memotong berdasarkan kepentingan data, dan pemotongan akan membatalkan pemformatan JSON.

- `payload`
- `awsRequest`
- `awsResponse`

Jika EventBridge tidak memotong bidang dalam acara tersebut, `truncatedFields` bidang tersebut menyertakan daftar bidang data terpotong.

Pelaporan kesalahan dalam catatan log EventBridge Pipes

EventBridge juga mencakup data kesalahan, jika tersedia, dalam langkah-langkah eksekusi pipa yang mewakili status kegagalan. Langkah-langkah ini meliputi:

- `ExecutionThrottled`
- `ExecutionTimeout`
- `ExecutionFailed`
- `ExecutionPartiallyFailed`
- `EnrichmentTransformationFailed`
- `EnrichmentInvocationFailed`
- `EnrichmentStageFailed`
- `TargetTransformationFailed`
- `TargetInvocationFailed`

- `TargetInvocationPartiallyFailed`
- `TargetStageFailed`
- `TargetStagePartiallyFailed`

EventBridge Langkah eksekusi pipa

Memahami alur langkah eksekusi pipa dapat membantu Anda dalam pemecahan masalah atau men-debug kinerja pipa Anda menggunakan log.

Eksekusi pipa adalah peristiwa atau kumpulan peristiwa yang diterima oleh pipa yang melakukan perjalanan ke pengayaan atau target. Jika diaktifkan, EventBridge buat catatan log untuk setiap langkah eksekusi yang dilakukannya saat batch acara diproses.

Pada tingkat tinggi, eksekusi berisi dua tahap, atau kumpulan langkah: pengayaan, dan target. Masing-masing tahapan ini terdiri dari langkah-langkah transformasi dan doa.

Langkah-langkah utama dari eksekusi pipa yang sukses mengikuti aliran ini:

- Eksekusi pipa dimulai.
- Eksekusi memasuki tahap pengayaan jika Anda telah menentukan pengayaan untuk acara tersebut. Jika Anda belum menentukan pengayaan, eksekusi berlanjut ke tahap target.

  Pada tahap pengayaan, pipa melakukan transformasi apa pun yang telah Anda tentukan, lalu memanggil pengayaan.
- Pada tahap target, pipa melakukan transformasi apa pun yang telah Anda tentukan, lalu memanggil target.

  Jika Anda belum menentukan transformasi atau target, eksekusi akan melewati tahap target.
- Eksekusi pipa selesai dengan sukses.

Diagram di bawah ini menunjukkan aliran ini. Jalur divergen diformat sebagai garis putus-putus.

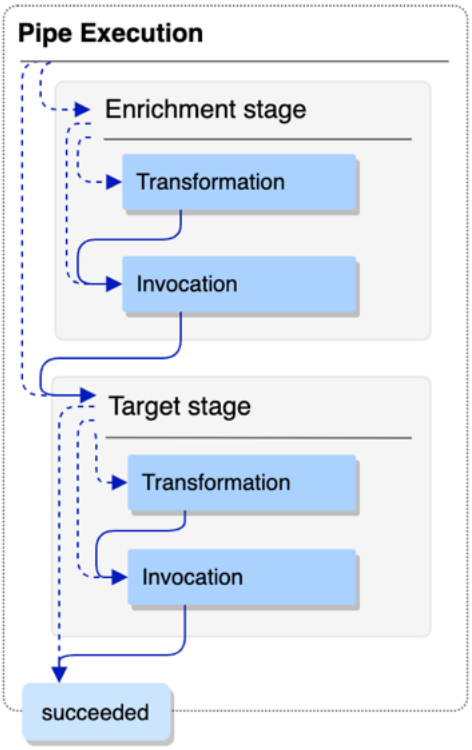
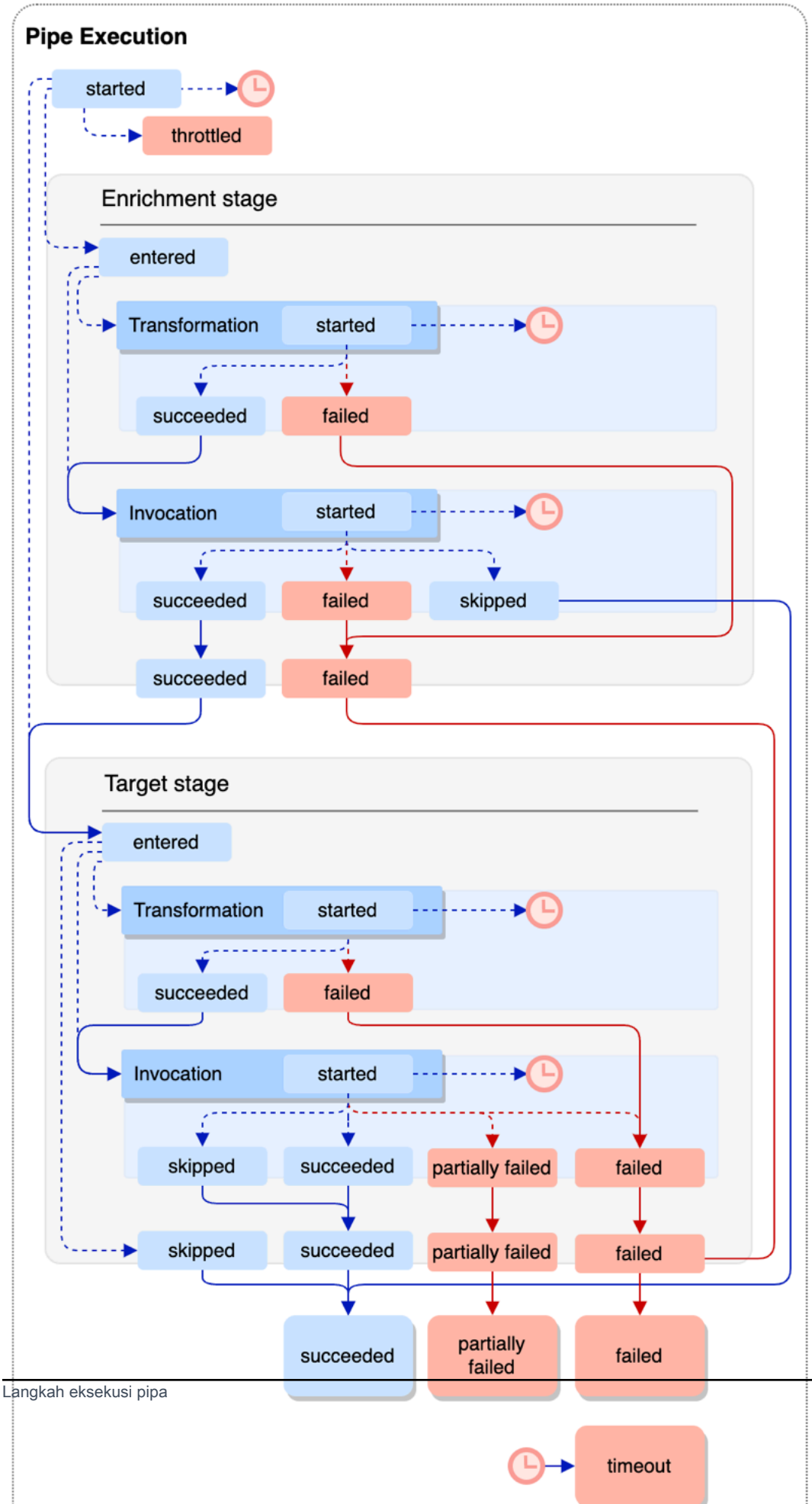


Diagram di bawah ini menyajikan tampilan rinci dari aliran eksekusi pipa, dengan semua langkah eksekusi yang mungkin diwakili. Sekali lagi, jalur divergen diformat sebagai garis putus-putus

Untuk daftar lengkap langkah eksekusi pipa, lihat???.


Pipe Execution
started
throttled
Enrichment stage
entered
Transformation
started
succeeded
failed
Invocation
started
succeeded
failed
skipped
succeeded
failed
Target stage
entered
Transformation
started
succeeded
failed
Invocation
started
skipped
succeeded
partially failed
failed
skipped
succeeded
partially failed
failed
succeeded
partially failed
failed
timeout

Perhatikan bahwa pemanggilan target dapat mengakibatkan kegagalan sebagian batch. Untuk informasi selengkapnya, lihat ???.

EventBridge Referensi skema log pipa

Referensi berikut merinci skema untuk catatan log EventBridge Pipa.

Setiap catatan log mewakili langkah eksekusi pipa, dan dapat berisi hingga 10.000 peristiwa jika sumber pipa dan target telah dikonfigurasi untuk batching.

Untuk informasi selengkapnya, lihat ???.

```
{
    "executionId": "guid",
    "timestamp": "date_time",
    "messageType": "execution_step",
    "resourceArn": "arn:aws:pipes:region:account:pipe/pipe-name",
    "logLevel": "TRACE | INFO | ERROR",
    "payload": "{}",
    "awsRequest": "{}"
    "awsResponse":"{}"
    "truncatedFields": ["awsRequest","awsResponse","payload"],
    "error": {
        "httpStatusCode": code,
        "message": "error_message",
        "details": "",
        "awsService": "service_name",
        "requestId": "service_request_id"
    }
}
```

ExecutionID

ID eksekusi pipa.

Eksekusi pipa adalah peristiwa atau kumpulan peristiwa yang diterima oleh pipa yang melakukan perjalanan ke pengayaan atau target. Untuk informasi selengkapnya, lihat ???.

timestamp

Tanggal dan waktu peristiwa log dipancarkan.

Satuan: milidetik

messageType

Langkah eksekusi pipa di mana catatan dihasilkan.

Untuk informasi selengkapnya tentang langkah-langkah eksekusi pipa, lihat???.

resourceArn

Nama Sumber Daya Amazon (ARN) untuk pipa.

logLevel

Tingkat detail yang ditentukan untuk log pipa.

Nilai yang valid: `ERROR` | `INFO` | `TRACE`

Untuk informasi selengkapnya, lihat ???.

payload

Isi batch acara sedang diproses oleh pipa.

EventBridge termasuk bidang ini hanya jika Anda telah menentukan untuk menyertakan data eksekusi dalam log untuk pipa ini. Lihat informasi yang lebih lengkap di ???

> **Important**
>
> Bidang ini mungkin berisi informasi sensitif. EventBridge tidak berusaha untuk menyunting konten bidang ini selama pencatatan.

Untuk informasi selengkapnya, lihat ???.

AWSRequest

Permintaan dikirim ke pengayaan atau target, dalam format JSON. Untuk permintaan yang dikirim ke tujuan API, ini mewakili permintaan HTTP yang dikirim ke titik akhir tersebut.

EventBridge termasuk bidang ini hanya jika Anda telah menentukan untuk menyertakan data eksekusi dalam log untuk pipa ini. Lihat informasi yang lebih lengkap di ???

> **Important**
>
> Bidang ini mungkin berisi informasi sensitif. EventBridge tidak berusaha untuk menyunting konten bidang ini selama pencatatan.

Untuk informasi selengkapnya, lihat ???.

AWSResponse

Respons dikembalikan oleh pengayaan atau target, dalam format JSON. Untuk permintaan yang dikirim ke tujuan API, ini mewakili respons HTTP yang dikembalikan dari titik akhir tersebut, dan bukan respons yang dikembalikan oleh layanan Tujuan API itu sendiri.

EventBridge termasuk bidang ini hanya jika Anda telah menentukan untuk menyertakan data eksekusi dalam log untuk pipa ini. Lihat informasi yang lebih lengkap di ???

> **Important**
>
> Bidang ini mungkin berisi informasi sensitif. EventBridge tidak berusaha untuk menyunting konten bidang ini selama pencatatan.

Untuk informasi selengkapnya, lihat ???.

Terpotong Fields

Daftar bidang data eksekusi EventBridge telah dipotong untuk menyimpan catatan di bawah batasan ukuran 256 KB.

Jika EventBridge tidak harus memotong salah satu bidang data eksekusi, bidang ini hadir tetapi. `null`

Untuk informasi selengkapnya, lihat ???.

kesalahan

Berisi informasi untuk setiap kesalahan yang dihasilkan selama langkah eksekusi pipa ini.

Jika tidak ada kesalahan yang dihasilkan selama langkah eksekusi pipa ini, bidang ini ada tetapi`null`.

httpStatusCode

Kode status HTTP dikembalikan oleh layanan yang disebut.

message

Pesan kesalahan dikembalikan oleh layanan yang disebut.

rincian

Setiap informasi kesalahan terperinci yang dikembalikan oleh layanan yang disebut.

AWSService

Nama layanan yang disebut.

RequesTid

ID permintaan untuk permintaan ini dari layanan yang disebut.

Pencatatan dan pemantauan EventBridge Pipa Amazon menggunakan AWS CloudTrail dan CloudWatch Log Amazon

Anda dapat mencatat pemanggilan EventBridge Pipa CloudTrail dan menggunakan serta memantau kesehatan pipa Anda menggunakan CloudWatch metrik.

CloudWatch metrik

EventBridge Pipes mengirimkan metrik ke Amazon CloudWatch setiap menit untuk segala hal mulai dari eksekusi pipa yang dibatasi hingga target yang berhasil dipanggil.

Metrik	Deskripsi
`Concurrency`	Jumlah eksekusi pipa secara bersamaan. Dimensi yang Valid: AwsAccountId Satuan: Tidak ada
`Duration`	Lama waktu eksekusi pipa berlangsung. Dimensi yang Valid: PipeName Unit: Milidetik
`EventCount`	Jumlah peristiwa yang telah diproses pipa. Dimensi yang Valid: PipeName Satuan: Tidak ada

Metrik	Deskripsi
`EventSize`	Ukuran muatan acara yang memanggil pipa. Dimensi yang Valid: PipeName Satuan: Byte
`Execution Throttled`	Berapa banyak eksekusi pipa yang dibatasi. **Note** Nilai ini akan menjadi `0` jika tidak ada eksekusi yang dibatasi. Dimensi yang Valid: AwsAccountId, PipeName Satuan: Tidak ada
`Execution Timeout`	Berapa banyak eksekusi pipa yang habis waktunya sebelum menyelesa ikan eksekusi. **Note** Nilai ini akan menjadi `0` jika tidak ada eksekusi yang habis waktunya. Dimensi yang Valid: PipeName Satuan: Tidak ada

Metrik	Deskripsi
`ExecutionFailed`	Berapa banyak eksekusi pipa yang gagal. **Note** Nilai ini akan terjadi `0` jika tidak ada eksekusi yang gagal. Dimensi yang Valid: PipeName Satuan: Tidak ada
`Execution Partially Failed`	Berapa banyak eksekusi pipa yang sebagian gagal. **Note** Nilai ini akan menjadi `0` jika tidak ada eksekusi sebagian gagal. Dimensi yang Valid: PipeName Satuan: Tidak ada
`Enrichmen tStageDuration`	Berapa lama tahap pengayaan selesai. Dimensi yang Valid: PipeName Unit: Milidetik
`Enrichmen tStageFailed`	Berapa banyak eksekusi tahap pengayaan pipa yang gagal. **Note** Nilai ini akan terjadi `0` jika tidak ada eksekusi yang gagal. Dimensi yang Valid: PipeName Satuan: Tidak ada

Metrik	Deskripsi
`Invocations`	Jumlah total pemanggilan. Dimensi yang Valid: AwsAccountId, PipeName Satuan: Tidak ada
`TargetStageDuration`	Berapa lama tahap target selesai. Dimensi yang Valid: PipeName Unit: Milidetik
`TargetStageFailed`	Berapa banyak eksekusi tahap target pipa yang gagal. **Note** Nilai ini akan terjadi `0` jika tidak ada eksekusi yang gagal. Dimensi yang Valid: PipeName Satuan: Tidak ada
`TargetStagePartiallyFailed`	Berapa banyak eksekusi tahap target pipa yang sebagian gagal. **Note** Nilai ini akan menjadi `0` jika tidak ada eksekusi tahap target yang gagal sebagian. Dimensi yang Valid: PipeName Satuan: Tidak ada

Metrik	Deskripsi
`TargetStageSkipped`	Berapa banyak eksekusi tahap target pipa yang dilewati (misalnya, karena pengayaan mengembalikan muatan kosong). Dimensi yang Valid: PipeName Unit: Hitungan

Dimensi untuk CloudWatch metrik

CloudWatch metrik memiliki dimensi, atau atribut yang dapat diurutkan, yang tercantum di bawah ini.

Dimensi	Deskripsi
`AwsAccountId`	Memfilter metrik yang tersedia berdasarkan ID akun.
`PipeName`	Memfilter metrik yang tersedia berdasarkan nama pipa.

AmazonEventBridgePenanganan kesalahan pipa dan pemecahan masalah

Coba lagi perilaku dan penanganan kesalahan

EventBridgePipa secara otomatis mencoba ulang pengayaan dan menargetkan pemanggilan pada setiap yang dapat dicoba ulangAWSkegagalan dengan layanan sumber, pengayaan atau layanan target, atauEventBridge. Namun, jika ada kegagalan yang dikembalikan oleh pengayaan atau implementasi target pelanggan, throughput pemungutan suara pipa secara bertahap akan mundur. Untuk kesalahan 4xx yang hampir terus menerus (seperti masalah otorisasi dengan IAM atau sumber daya yang hilang), pipa dapat dinonaktifkan secara otomatis dengan pesan penjelasan di`StateReason`.

Kesalahan pemanggilan pipa dan perilaku coba lagi

Saat Anda memanggil pipa, dua jenis kesalahan utama dapat terjadi:kesalahan internal pipadankesalahan pemanggilan pelanggan.

Kesalahan internal pipa

Kesalahan internal pipa adalah kesalahan yang dihasilkan oleh aspek pemanggilan yang dikelola olehEventBridgeLayanan pipa.

Jenis kesalahan ini dapat mencakup masalah seperti:

- Kegagalan koneksi HTTP saat mencoba memanggil layanan targer pelanggan
- Penurunan ketersediaan sementara pada layanan pipa itu sendiri.

Secara umum,EventBridgePipa mencoba ulang kesalahan internal dalam jumlah yang tidak terbatas, dan berhenti hanya ketika catatan kedaluwarsa di sumbernya.

Untuk pipa dengan sumber aliran,EventBridgePipa tidak menghitung percobaan ulang untuk kesalahan internal terhadap jumlah maksimum percobaan ulang yang ditentukan pada kebijakan coba lagi untuk sumber aliran. Untuk pipa dengan sumber Amazon SQS,EventBridgePipa tidak menghitung percobaan ulang untuk kesalahan internal terhadap jumlah penerimaan maksimum untuk sumber Amazon SQS.

Kesalahan pemanggilan pelanggan

Kesalahan pemanggilan pelanggan adalah kesalahan yang dihasilkan dari konfigurasi atau kode yang dikelola oleh pengguna.

Jenis kesalahan ini dapat mencakup masalah seperti:

- Izin yang tidak memadai pada pipa untuk memanggil target.
- Kesalahan logika di Lambda pelanggan yang dipanggil secara sinkron, Fungsi Langkah, tujuan API, atau titik akhir API Gateway.

Untuk kesalahan pemanggilan pelanggan,EventBridgePipa melakukan hal berikut:

- Untuk pipa dengan sumber aliran,EventBridgePipa mencoba ulang hingga waktu percobaan ulang maksimum yang dikonfigurasi pada kebijakan coba lagi pipa atau sampai usia rekor maksimum berakhir, mana yang lebih dulu.
- Untuk pipa dengan sumber Amazon SQS,EventBridgePipa mencoba ulang kesalahan pelanggan hingga jumlah penerimaan maksimum pada antrian sumber.

- Untuk pipa dengan sumber Apache Kafka atau Amazon MQ,EventBridgemencoba ulang kesalahan pelanggan sama seperti mencoba ulang kesalahan internal.

Untuk pipa dengan target komputasi, Anda harus memanggil pipa secara serempak agarEventBridgePipa untuk mengetahui kesalahan runtime yang dilemparkan dari logika komputasi pelanggan dan mencoba lagi kesalahan tersebut. Pipa tidak dapat mencoba lagi kesalahan yang dilemparkan dari logika alur kerja standar Step Functions, karena target ini harus dipanggil secara asinkron.

Untuk Amazon SQS dan sumber streaming, seperti Kinesis dan DynamoDB,EventBridgePipa mendukung penanganan kegagalan batch sebagian dari kegagalan target. Untuk informasi lebih lanjut, lihatKegagalan batch sebagian.

Perilaku pipa DLQ

Pipa mewarisi perilaku antrian huruf mati (DLQ) dari sumbernya:

- Jika antrian Amazon SQS sumber memiliki DLQ yang dikonfigurasi, pesan secara otomatis dikirim ke sana setelah jumlah upaya yang ditentukan.
- Untuk sumber aliran, seperti aliran DynamoDB dan Kinesis, Anda dapat mengonfigurasi DLQ untuk peristiwa pipa dan rute. Sumber aliran DynamoDB dan Kinesis mendukung antrian Amazon SQS dan topik Amazon SNS sebagai target DLQ.

Jika Anda menentukan`DeadLetterConfig`untuk pipa dengan sumber Kinesis atau DynamoDB, pastikan bahwa`MaximumRecordAgeInSeconds`properti pada pipa kurang dari`MaximumRecordAge`dari acara sumber.`MaximumRecordAgeInSeconds`mengontrol kapan poller pipa akan menyerah pada acara tersebut dan mengirimkannya ke DLQ dan`MaximumRecordAge`mengontrol berapa lama pesan akan terlihat di aliran sumber sebelum dihapus. Oleh karena itu, atur`MaximumRecordAgeInSeconds`dengan nilai yang kurang dari sumbernya`MaximumRecordAge`sehingga ada waktu yang cukup antara saat acara dikirim ke DLQ, dan kapan acara itu dihapus secara otomatis oleh sumber bagi Anda untuk menentukan mengapa acara tersebut masuk ke DLQ.

Untuk sumber Amazon MQ, DLQ dapat dikonfigurasi langsung di broker pesan.

EventBridgePipa tidak mendukung DLQ first-in first-out (FIFO) untuk sumber aliran.

EventBridgePipes tidak mendukung DLQ untuk aliran MSK Amazon dan sumber aliran Apache Kafka yang dikelola sendiri.

Status kegagalan pipa

Membuat, menghapus, dan memperbarui pipa adalah operasi asinkron yang dapat mengakibatkan status kegagalan. Demikian juga, pipa mungkin dinonaktifkan secara otomatis karena kegagalan. Dalam semua kasus, pipa`StateReason`memberikan informasi untuk membantu memecahkan masalah kegagalan.

Berikut ini adalah contoh dari kemungkinan`StateReason`nilai:

- Stream tidak ditemukan. Untuk melanjutkan pemrosesan, hapus pipa dan buat yang baru.
- Pipa tidak memiliki izin yang diperlukan untuk melakukan operasi Antrian (sqs:ReceiveMessage, sqs:DeleteMessagedan sqs:GetQueueAttributes)
- Kesalahan koneksi. VPC Anda harus dapat terhubung ke pipa. Anda dapat memberikan akses dengan mengonfigurasi Gateway NAT. Untuk cara mengatur gateway NAT, silakan periksaAWSdokumentasi.
- Kluster MSK tidak memiliki grup keamanan yang terkait dengannya

Pipa dapat dihentikan secara otomatis dengan yang diperbarui`StateReason`. Alasan yang mungkin termasuk:

- Alur kerja standar Fungsi Langkah yang dikonfigurasi sebagaipengayaan.
- Alur kerja standar Fungsi Langkah yang dikonfigurasi sebagai targetdipanggil secara sinkron.

Kegagalan enkripsi khusus

Jika Anda mengonfigurasi sumber untuk menggunakanAWS KMSkunci enkripsi khusus (CMK), bukanAWS-dikelolaAWS KMSkunci, Anda harus secara eksplisit memberikan izin dekripsi Peran Eksekusi pipa Anda. Untuk melakukannya, sertakan izin tambahan berikut dalam kebijakan CMK khusus:

```
{
    "Sid": "Allow Pipes access",
    "Effect": "Allow",
    "Principal": {
```

```
          "AWS": "arn:aws:iam::01234567890:role/service-role/
Amazon_EventBridge_Pipe_DDBStreamSourcePipe_12345678"
      },
      "Action": "kms:Decrypt",
      "Resource": "*"
  }
```

Ganti peran di atas dengan Peran Eksekusi pipa Anda.

Hal ini berlaku untuk semua sumber pipa denganAWS KMSCMK, termasuk:

- Amazon DynamoDB Streams
- Amazon Kinesis Data Streams
- Amazon MQ
- Amazon MSK
- Amazon SQS

Tutorial: Buat EventBridge pipa yang memfilter peristiwa sumber

Dalam tutorial ini, Anda akan membuat pipa yang menghubungkan sumber aliran DynamoDB ke target antrian Amazon SQS. Ini termasuk menentukan pola peristiwa untuk pipa yang akan digunakan saat memfilter peristiwa untuk dikirim ke antrian. Anda kemudian akan menguji pipa untuk memastikan bahwa hanya acara yang diinginkan yang dikirimkan.

Prasyarat: Buat sumber dan target

Sebelum Anda membuat pipa, Anda harus membuat sumber dan menargetkan bahwa pipa akan terhubung. Dalam hal ini, aliran data Amazon DynamoDB bertindak sebagai sumber pipa, dan antrian Amazon SQS sebagai target pipa.

Untuk menyederhanakan langkah ini, Anda dapat menggunakan AWS CloudFormation untuk menyediakan sumber dan sumber daya target. Untuk melakukan ini, Anda akan membuat CloudFormation template yang mendefinisikan sumber daya berikut:

- Sumber pipa

  Tabel Amazon DynamoDB, `pipe-tutorial-source` bernama, dengan aliran diaktifkan untuk menyediakan aliran informasi yang diurutkan tentang perubahan item dalam tabel DynamoDB.

- Target pipa

  Antrian Amazon SQS, bernama`pipe-tutorial-target`, untuk menerima aliran peristiwa DynamoDB dari pipa Anda.

Untuk membuat CloudFormation template untuk penyediaan sumber daya pipa

1. Salin teks template JSON di ??? bagian, di bawah ini.
2. Simpan template sebagai file JSON (misalnya,`~/pipe-tutorial-resources.json`).

Selanjutnya, gunakan file template yang baru saja Anda buat untuk menyediakan CloudFormation tumpukan.

> **Note**
>
> Setelah Anda membuat CloudFormation tumpukan Anda, Anda akan dikenakan biaya untuk AWS sumber daya yang disediakannya.

Menyediakan prasyarat tutorial menggunakan CLI AWS

- Jalankan perintah CLI berikut, di mana `--template-body` menentukan lokasi file template Anda:

```
aws cloudformation create-stack --stack-name pipe-tuturial-resources --template-
body file://~/pipe-tutorial-resources.json
```

Ketentuan prasyarat tutorial menggunakan konsol CloudFormation

1. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
2. Pilih Tumpukan, lalu pilih Buat tumpukan, dan pilih dengan sumber daya baru (standar).

   CloudFormation menampilkan wizard Buat tumpukan.
3. Untuk Prasyarat - Siapkan template, biarkan default, Template siap, dipilih.
4. Di bawah Tentukan templat, pilih Unggah file templat, lalu pilih file dan pilih Berikutnya.
5. Konfigurasikan tumpukan dan sumber daya yang akan disediakannya:

- Untuk nama Stack, masukkan`pipe-tuturial-resources`.
- Untuk Parameter, tinggalkan nama default untuk tabel DynamoDB dan antrian Amazon SQS.
- Pilih Selanjutnya.

6. Pilih Berikutnya, lalu pilih Kirim.

   CloudFormation membuat tumpukan dan menyediakan sumber daya yang ditentukan dalam template.

Untuk informasi lebih lanjut tentang CloudFormation, lihat Apa ituAWS CloudFormation? dalam AWS CloudFormationUser Guide.

Langkah 1: Buat pipa

Dengan sumber pipa dan target yang disediakan, Anda sekarang dapat membuat pipa untuk menghubungkan dua layanan.

Buat pipa menggunakan EventBridge konsol

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Pada panel navigasi, pilih Pipa.
3. Pilih Buat pipa.
4. Untuk Nama, beri nama pipa Andapipe-tutorial.
5. Tentukan sumber aliran data DynamoDB:

   a. Di bawah Detail, untuk Sumber, pilih aliran data DynamoDB.

      EventBridge menampilkan pengaturan konfigurasi sumber khusus DynamoDB.

   b. Untuk aliran DynamoDB, pilih. `pipe-tutorial-source`

      Biarkan posisi Mulai diatur ke default,`Latest`.

   c. Pilih Selanjutnya.

6. Tentukan dan uji pola acara untuk memfilter peristiwa:

   Penyaringan memungkinkan Anda untuk mengontrol peristiwa mana yang dikirim pipa ke pengayaan atau target. Pipa hanya mengirimkan peristiwa yang cocok dengan pola acara ke pengayaan atau target.

Untuk informasi selengkapnya, lihat ???.

> **Note**
>
> Anda hanya ditagih untuk acara-acara yang dikirim ke pengayaan atau target.

a. Di bawah Contoh acara - opsional, biarkan AWSacara dipilih, dan pastikan bahwa DynamoDB Stream Sample event 1 dipilih.

   Ini adalah contoh acara yang akan Anda gunakan untuk menguji pola acara kami.

b. Di bawah pola Peristiwa, masukkan pola acara berikut:

```
{
  "eventName": ["INSERT", "MODIFY"]
}
```

c. Pilih Pola uji.

   EventBridge menampilkan pesan bahwa peristiwa sampel cocok dengan pola acara. Ini karena peristiwa sampel memiliki `eventName` nilai`INSERT`.

d. Pilih Selanjutnya.

7. Pilih Berikutnya untuk melewatkan menentukan pengayaan.

   Dalam contoh ini, Anda tidak akan memilih pengayaan. Pengayaan memungkinkan Anda memilih layanan untuk meningkatkan data dari sumber sebelum mengirimkannya ke target. Untuk rincian lebih lanjut, lihat ???.

8. Tentukan antrean Amazon SQS Anda sebagai target pipa:

   a. Di bawah Detail, untuk layanan Target, pilih antrean Amazon SQS.

   b. Untuk Antrian, pilih`pipe-tutorial-target`.

   c. Biarkan bagian transformator Input Target kosong.

      Untuk informasi selengkapnya, lihat ???.

9. Pilih Buat Pipa

   EventBridge membuat pipa dan menampilkan halaman detail pipa. Pipa siap setelah statusnya diperbarui ke`Running`.

Langkah 2: Konfirmasikan peristiwa filter pipa

Pipa diatur, tetapi belum menerima acara dari meja.

Untuk menguji pipa, Anda akan memperbarui entri dalam tabel DynamoDB. Setiap pembaruan akan menghasilkan peristiwa yang dikirim aliran DynamoDB ke pipa kami. Beberapa akan cocok dengan pola acara yang Anda tentukan, beberapa tidak. Anda kemudian dapat memeriksa antrean Amazon SQS untuk memastikan bahwa pipa hanya mengirimkan peristiwa yang cocok dengan pola acara kami.

Perbarui item tabel untuk menghasilkan acara

1. Buka konsol DynamoDB di https://console.aws.amazon.com/dynamodb/.
2. Dari navigasi kiri, pilih Tabel. Pilih `pipe-tutorial-source` tabel.

   DynamoDB menampilkan halaman rincian tabel untuk. `pipe-tutorial-source`
3. Pilih Jelajahi item tabel, lalu pilih Buat item.

   DynamoDB menampilkan halaman Create item.
4. Di bawah Atribut, buat item tabel baru:

   a. Untuk Album masuk`Album A`.
   b. Untuk Artis masuk`Artist A`.
   c. Pilih Buat item.
5. Perbarui item tabel:

   a. Di bawah Item yang dikembalikan, pilih Album A.
   b. Pilih Tambahkan atribut baru, lalu pilih String.
   c. Masukkan nilai baru`Song`, dengan nilai`Song A`.
   d. Pilih Save changes (Simpan perubahan).
6. Hapus item tabel:

   a. Di bawah Item yang dikembalikan, periksa Album A.
   b. Dari menu Tindakan, pilih Hapus item.

Anda telah membuat tiga pembaruan pada item tabel; ini menghasilkan tiga peristiwa untuk aliran data DynamoDB:

- `INSERT`Peristiwa saat Anda membuat item.
- Sebuah `MODIFY` peristiwa ketika Anda menambahkan atribut ke item.
- `REMOVE`Peristiwa saat Anda menghapus item.

Namun, pola acara yang Anda tentukan untuk pipa harus menyaring setiap peristiwa yang bukan `INSERT` atau `MODIFY` peristiwa. Selanjutnya, konfirmasikan bahwa pipa mengirimkan peristiwa yang diharapkan ke antrian.

Konfirmasikan peristiwa yang diharapkan telah dikirimkan ke antrian

1. Buka konsol Amazon SQS di https://console.aws.amazon.com/sqs/.
2. Pilih `pipe-tutorial-target` antrian.

   Amazon SQS menampilkan halaman detail antrian.
3. Pilih Kirim dan terima pesan, lalu di bawah Menerima pesan pilih Poll untuk pesan.

   Antrian polling pipa dan kemudian daftar peristiwa yang diterimanya.
4. Pilih nama acara untuk melihat acara JSON yang telah disampaikan.

Harus ada dua peristiwa dalam antrian: satu dengan `eventName` dari`INSERT`, dan satu dengan `eventName` dari. `MODIFY` Namun, pipa tidak mengirimkan acara untuk menghapus item tabel, karena peristiwa itu memiliki `eventName` of`REMOVE`, yang tidak cocok dengan pola acara yang Anda tentukan dalam pipa.

Langkah 3: Bersihkan sumber daya Anda

Pertama, hapus pipa itu sendiri.

Hapus pipa menggunakan EventBridge konsol

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Pada panel navigasi, pilih Pipa.
3. Pilih `pipe-tutorial` pipa, dan pilih Hapus.

Kemudian, hapus CloudFormation tumpukan, untuk mencegah ditagih untuk penggunaan berkelanjutan dari sumber daya yang disediakan di dalamnya.

Hapus prasyarat tutorial menggunakan CLI AWS

- Jalankan perintah CLI berikut, di mana `--stack-name` menentukan nama tumpukan Anda:

```
aws cloudformation delete-stack --stack-name pipe-tuturial-resources
```

Hapus prasyarat tutorial menggunakan konsol AWS CloudFormation

1. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
2. Pada halaman Stacks, pilih tumpukan dan kemudian pilih Hapus.
3. Pilih Hapus untuk mengonfirmasi tindakan Anda.

AWS CloudFormationtemplate untuk menghasilkan prasyarat

Gunakan JSON di bawah ini untuk membuat CloudFormation template untuk menyediakan sumber dan sumber daya target yang diperlukan untuk tutorial ini.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",

 "Description" : "Provisions resources to use with the EventBridge Pipes tutorial. You
 will be billed for the AWS resources used if you create a stack from this template.",

  "Parameters" : {
    "SourceTableName" : {
      "Type" : "String",
      "Default" : "pipe-tutorial-source",
      "Description" : "Specify the name of the table to provision as the pipe source,
 or accept the default."
    },
  "TargetQueueName" : {
    "Type" : "String",
    "Default" : "pipe-tutorial-target",
    "Description" : "Specify the name of the queue to provision as the pipe target, or
 accept the default."
  }
},
  "Resources": {
    "PipeTutorialSourceDynamoDBTable": {
      "Type": "AWS::DynamoDB::Table",
```

```
      "Properties": {
        "AttributeDefinitions": [{
            "AttributeName": "Album",
            "AttributeType": "S"
          },
          {
            "AttributeName": "Artist",
            "AttributeType": "S"
          }

        ],
        "KeySchema": [{
            "AttributeName": "Album",
            "KeyType": "HASH"

          },
          {
            "AttributeName": "Artist",
            "KeyType": "RANGE"
          }
        ],
        "ProvisionedThroughput": {
          "ReadCapacityUnits": 10,
          "WriteCapacityUnits": 10
        },
        "StreamSpecification": {
          "StreamViewType": "NEW_AND_OLD_IMAGES"
        },
        "TableName": { "Ref" : "SourceTableName" }
      }
    },
    "PipeTutorialTargetQueue": {
      "Type": "AWS::SQS::Queue",
      "Properties": {
        "QueueName": { "Ref" : "TargetQueueName" }
      }
    }
  }
}
```

Menghasilkan AWS CloudFormation template dari EventBridge Pipes

AWS CloudFormation memungkinkan Anda mengonfigurasi dan mengelola AWS sumber daya Anda di seluruh akun dan wilayah secara terpusat dan berulang dengan memperlakukan infrastruktur sebagai kode. CloudFormation melakukan ini dengan membiarkan Anda membuat template, yang menentukan sumber daya yang ingin Anda sediakan dan kelola.

EventBridge memungkinkan Anda untuk membuat template dari pipa yang ada di akun Anda, sebagai bantuan untuk membantu Anda mulai mengembangkan CloudFormation template. Anda dapat memilih satu pipa, atau beberapa pipa untuk disertakan dalam template. Anda kemudian dapat menggunakan template ini sebagai dasar untuk membuat tumpukan sumber daya yang CloudFormation dikelola.

Untuk informasi selengkapnya CloudFormation, lihat Panduan AWS CloudFormation Pengguna.

Untuk bus acara, Anda dapat membuat CloudFormation template dari bus acara dan aturan bus acara.

Sumber daya termasuk dalam template EventBridge Pipa

Ketika EventBridge menghasilkan CloudFormation template, itu menciptakan AWS::Pipes::Pipesumber daya untuk setiap pipa yang dipilih. Selain itu, EventBridge termasuk sumber daya berikut di bawah kondisi yang dijelaskan:

- AWS::Events::ApiDestination

  Jika pipa Anda menyertakan tujuan API, baik sebagai pengayaan atau target, EventBridge sertakan dalam CloudFormation template sebagai AWS::Events::ApiDestination sumber daya.
- AWS::Events::EventBus

  Jika pipa Anda menyertakan bus acara sebagai target, EventBridge sertakan dalam CloudFormation templat sebagai AWS::Events::EventBus sumber daya.
- AWS::IAM::Role

  Jika Anda telah EventBridge membuat peran eksekusi baru saat mengonfigurasi pipa, Anda dapat memilih untuk EventBridge menyertakan peran tersebut dalam templat sebagai AWS::IAM::Role sumber daya. EventBridge tidak termasuk peran yang Anda buat. (Dalam kedua kasus, `RoleArn` properti AWS::Pipes::Pipe sumber daya berisi ARN peran.)

Pertimbangan saat menggunakan CloudFormation template yang dihasilkan dari Pipes EventBridge

Pertimbangkan faktor-faktor berikut saat menggunakan CloudFormation templat yang Anda hasilkan EventBridge:

- EventBridge tidak menyertakan kata sandi apa pun dalam template generate.

  Anda dapat mengedit templat untuk menyertakan parameter templat yang memungkinkan pengguna menentukan kata sandi atau informasi sensitif lainnya saat menggunakan templat untuk membuat atau memperbarui CloudFormation tumpukan.

  Selain itu, pengguna dapat menggunakan Secrets Manager untuk membuat rahasia di wilayah yang diinginkan dan kemudian mengedit template yang dihasilkan untuk menggunakan parameter dinamis.
- Target dalam template yang dihasilkan tetap persis seperti yang ditentukan dalam pipa asli. Hal ini dapat menyebabkan masalah lintas wilayah jika Anda tidak mengedit template dengan tepat sebelum menggunakannya untuk membuat tumpukan di wilayah lain.

  Selain itu, template yang dihasilkan tidak membuat target hilir secara otomatis.

Menghasilkan CloudFormation template dari EventBridge Pipes

Untuk menghasilkan CloudFormation template dari satu atau lebih pipa menggunakan EventBridge konsol, lakukan hal berikut:

Untuk menghasilkan CloudFormation template dari satu atau lebih pipa

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Pipa.
3. Di bawah Pipa, pilih satu atau lebih pipa yang ingin Anda sertakan dalam CloudFormation templat yang dihasilkan.

   Untuk satu pipa, Anda juga dapat memilih nama pipa untuk menampilkan halaman detail pipa.
4. Pilih CloudFormation Template, lalu pilih format mana yang EventBridge ingin Anda buat template di: JSON atau YAMG.

   EventBridge menampilkan template, yang dihasilkan dalam format yang dipilih.

5. Jika Anda telah EventBridge membuat peran eksekusi baru untuk salah satu pipa yang dipilih, dan Anda EventBridge ingin menyertakan peran tersebut dalam template, pilih Sertakan IAM peran yang dibuat oleh konsol atas nama Anda.
6. EventBridge memberi Anda pilihan untuk mengunduh file template, atau menyalin template ke clipboard.

   - Untuk mengunduh file templat, pilih Unduh.
   - Untuk menyalin template ke clipboard, pilih Salin.
7. Untuk keluar dari template, pilih Batal.

Membuat aplikasi toleransi patahan daerah dengan endpoint global dan replikasi peristiwa

Anda dapat meningkatkan ketersediaan aplikasi Anda dengan Amazon EventBridge titik akhir global. Endpoint global membantu membuat aplikasi Anda toleran terhadap kesalahan regional tanpa biaya tambahan. Untuk memulai, Anda menetapkan pemeriksaan kondisi Amazon Route 53 ke titik akhir. Ketika failover dimulai, pemeriksaan kesehatan melaporkan keadaan “tidak sehat”. Dalam beberapa menit dari inisiasi failover, semua kustomacaradialihkan keBus peristiwadi Wilayah sekunder dan diproses oleh bus acara tersebut. Setelah pemeriksaan kesehatan melaporkan keadaan “sehat”, acara diproses oleh bus acara di Wilayah utama.

Bila Anda menggunakan endpoint global, Anda dapat mengaktifkanReplikasi peristiwa. Replikasi acara mengirimkan semua acara khusus ke bus acara di Wilayah primer dan sekunder menggunakan aturan terkelola.

> **Note**
>
> Jika Anda menggunakan bus khusus, Anda memerlukan bus khusus di setiap Wilayah dengan nama yang sama dan di akun yang sama agar failover berfungsi dengan baik.

Topik

- Waktu Pemulihan & Tujuan Titik Pemulihan
- Replikasi peristiwa
- Membuat titik akhir global
- Bekerja dengan endpoint global dengan menggunakanAWSSDK
- Wilayah yang Tersedia
- Praktik terbaik untuk bekerja dengan Amazon EventBridge titik akhir global
- AWS CloudFormationtemplate untuk mengatur pemeriksaan kondisi Route 53

Waktu Pemulihan & Tujuan Titik Pemulihan

The Recovery Time Objective (RTO) adalah waktu yang dibutuhkan untuk Wilayah sekunder untuk mulai menerima peristiwa setelah kegagalan. Untuk RTO, waktu termasuk jangka waktu untuk

memicu CloudWatch alarm dan memperbarui status untuk pemeriksaan kondisi Route 53. The Recovery Point Objective (RPO) adalah ukuran data yang akan dibiarkan belum diproses selama kegagalan. Untuk RPO, waktu termasuk peristiwa yang tidak direplikasi ke Wilayah sekunder dan terjebak di Wilayah utama sampai layanan atau Wilayah pulih. Dengan endpoint global, jika Anda mengikuti panduan preskriptif kami untuk konfigurasi alarm, Anda dapat mengharapkan RTO dan RPO menjadi 360 detik dengan maksimum 420 detik.

Replikasi peristiwa

Acara diproses di Wilayah sekunder secara asinkron. Ini berarti bahwa peristiwa tidak dijamin akan diproses pada waktu yang sama di kedua Daerah. Ketika failover dipicu, peristiwa diproses oleh Wilayah sekunder dan akan diproses oleh Wilayah utama ketika tersedia. Mengaktifkan replikasi acara akan meningkatkan biaya bulanan Anda. Untuk informasi selengkapnya, lihatHarga Amazon EventBridge

Sebaiknya aktifkan replikasi peristiwa saat menyiapkan endpoint global karena alasan berikut:

- Replikasi acara membantu Anda memverifikasi bahwa endpoint global Anda dikonfigurasi dengan benar. Hal ini membantu untuk memastikan bahwa Anda akan dibahas dalam hal failover.
- Replikasi acara diperlukan untuk secara otomatis pulih dari peristiwa failover. Jika Anda tidak mengaktifkan replikasi acara, Anda harus mengatur ulang pemeriksaan kesehatan Route 53 secara manual ke “sehat” sebelum acara akan kembali ke Wilayah utama.

Muatan peristiwa

Berikut ini adalah contoh dari muatan peristiwa yang direplikasi:

> **Note**
>
> Untuk`region`, Wilayah yang acara direplikasi dari terdaftar.

```
{
    "version": "0",
    "id": "a908baa3-65e5-ab77-367e-527c0e71bbc2",
    "detail-type": "Test",
    "source": "test.service.com",
    "account": "0123456789",
```

```
    "time": "1900-01-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:events:us-east-1:0123456789:endpoint/MyEndpoint"
    ],
    "detail": {
        "a": "b"
    }
}
```

Membuat titik akhir global

Selesaikan langkah-langkah berikut untuk menyiapkan titik akhir global:

1. Pastikan bahwa Anda memiliki bus acara yang cocok dan aturan di kedua Wilayah primer dan sekunder.
2. BuatPemeriksaan kondisi Route 53untuk memantau bus acara Anda. Untuk bantuan dalam membuat pemeriksaan kesehatan Anda, pilihPemeriksaan Health Barusaat membuat endpoint global Anda.
3. Buat endpoint global Anda.

Setelah Anda mengatur pemeriksaan kesehatan Route 53, Anda dapat membuat endpoint global.

Untuk membuat titik akhir global menggunakan konsol

1. Buka Amazon EventBridge konsol dihttps://console.aws.amazon.com/events/.
2. Di panel navigasi, pilihTitik Akhir Global.
3. Pilih Buat Titik Akhir.
4. Masukkan nama dan deskripsi untuk titik akhir.
5. UntukBus acara di Wilayah utama, pilih bus acara yang Anda inginkan titik akhir yang terkait dengan.
6. UntukWilayah Sekunder, pilih Wilayah yang ingin Anda arahkan acara jika terjadi failover.

> **Note**
>
> ParameterBus acara di Wilayah sekunderdiisi otomatis dan tidak dapat diedit.

7. UntukPemeriksaan kondisi Route 53, pilih pemeriksaan kesehatan yang akan dipantau endpoint. Jika belum memiliki pemeriksaan kondisi, pilihPemeriksaan Health Baruuntuk membukaAWS CloudFormationkonsol dan membuat pemeriksaan kesehatan menggunakan CloudFormation templat.

   > **Note**
   >
   > Data yang hilang akan menyebabkan pemeriksaan kesehatan gagal. Jika Anda hanya perlu mengirim acara sebentar-sebentar, pertimbangkan untuk menggunakan yang lebih lamaMinimumEvaluationPeriode, atau memperlakukan data yang hilang sebagai 'hilang' bukan 'pelanggaran'.

8. (Opsional) UntukReplikasi peristiwamelakukan hal berikut:

   a. PilihReplikasi peristiwa.

   b. UntukPeran eksekusi, pilih apakah akan membuat yang baruAWS Identity and Access Managementperan atau menggunakan yang sudah ada. Lakukan hal berikut:

      - Pilih Membuat peran baru untuk sumber daya khusus ini. Opsional, Anda dapat memperbaruiNama peranuntuk membuat peran baru.
      - PilihMenggunakan peran yang ada. Kemudian, untukPeran eksekusi, pilih peran yang diinginkan untuk digunakan.

9. Pilih Create (Buat).

Untuk membuat titik akhir global menggunakan API

Untuk membuat titik akhir global menggunakan EventBridge API, lihatCreateEndPointdi Amazon EventBridge Referensi API.

Untuk membuat titik akhir global menggunakanAWS CloudFormation

Untuk membuat titik akhir global menggunakanAWS CloudFormationAPI, lihatAWS:: EventsdiAWS CloudFormationPanduan Pengguna.

Bekerja dengan endpoint global dengan menggunakanAWSSDK

> **Note**
>
> Support untuk C ++ akan segera hadir.

Saat menggunakanAWSSDK untuk bekerja dengan titik akhir global, perhatikan hal berikut:

- Anda harus memilikiAWSPerpustakaan umum Runtime (CRT) diinstal untuk SDK spesifik Anda. Jika Anda tidak menginstal CRT, Anda akan mendapatkan pesan pengecualian yang menunjukkan apa yang perlu diinstal. Untuk informasi selengkapnya, lihat yang berikut:
  - AWSPerpustakaan umum Runtime (CRT)
  - awslabs/aws-crt-java
  - awslabs/aws-crt-nodejs
  - awslabs/aws-crt-python
- Setelah Anda membuat titik akhir global, Anda harus menambahkan`endpointId`dan`EventBusName`untuk setiap`PutEvents`panggilan yang Anda gunakan.
- Endpoint global mendukung Signature Versi 4A. Versi Sigv4 ini memungkinkan permintaan untuk ditandatangani untuk beberapaWilayah AWS. Hal ini berguna dalam operasi API yang dapat mengakibatkan akses data dari salah satu dari beberapa Wilayah. Saat menggunakanAWSSDK, Anda menyediakan kredensia Anda dan permintaan ke endpoint global akan menggunakan Signature Version 4A tanpa konfigurasi tambahan. Untuk informasi lebih lanjut tentang Sigv4a, lihatPenandatangananAWSPermintaan APIdiAWSReferensi Umum.

Wilayah yang Tersedia

Wilayah berikut mendukung endpoint global:

- AS Timur (N. Virginia)
- US East (Ohio)
- US West (N. California)
- US West (Oregon)
- Canada (Central)

- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)
- Europe (Milan)
- Europe (Paris)
- Europe (Stockholm)
- Asia Pacific (Mumbai)
- Asia Pacific (Osaka)
- Asia Pacific (Seoul)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- South America (São Paulo)

Praktik terbaik untuk bekerja dengan Amazon EventBridge titik akhir global

Praktik terbaik berikut direkomendasikan saat Anda menyiapkan endpoint global.

Topik

- Mengaktifkan replikasi peristiwa
- Mencegah throttling peristiwa
- Menggunakan metrik pelanggan di pemeriksaan kondisi Amazon Route 53

Mengaktifkan replikasi peristiwa

Kami sangat menyarankan agar Anda mengaktifkan replikasi dan memproses acara Anda di Wilayah sekunder yang Anda tetapkan ke titik akhir global Anda. Ini memastikan bahwa aplikasi Anda di Wilayah sekunder dikonfigurasi dengan benar. Anda juga harus mengaktifkan replikasi untuk memastikan pemulihan otomatis ke Wilayah utama setelah masalah dikurangi.

ID peristiwa dapat berubah di seluruh panggilan API sehingga menghubungkan peristiwa di seluruh Wilayah mengharuskan Anda untuk memiliki pengenal unik yang tidak berubah. Konsumen juga

harus dirancang dengan idempotency dalam pikiran. Dengan begitu, jika Anda mereplikasi acara, atau memutarnya ulang dari arsip, tidak ada efek samping dari peristiwa yang sedang diproses di kedua Wilayah.

Mencegah throttling peristiwa

Untuk mencegah terjadinya terjadinya throttled, sebaiknya perbarui`PutEvents`dan target batas sehingga mereka konsisten di seluruh Wilayah.

Menggunakan metrik pelanggan di pemeriksaan kondisi Amazon Route 53

Hindari memasukkan metrik pelanggan dalam pemeriksaan kondisi Amazon Route 53. Termasuk metrik ini dapat menyebabkan penerbit Anda gagal ke Wilayah sekunder jika pelanggan mengalami masalah meskipun semua pelanggan lain tetap sehat di Wilayah utama. Jika salah satu subcriber Anda gagal memproses peristiwa di Wilayah utama, Anda harus mengaktifkan replikasi untuk memastikan bahwa pelanggan Anda di Wilayah sekunder dapat memproses peristiwa dengan sukses.

AWS CloudFormationtemplate untuk mengatur pemeriksaan kondisi Route 53

Saat menggunakan endpoint global Anda harus memiliki pemeriksaan kesehatan Route 53 untuk memantau status Wilayah Anda. Template berikut mendefinisikanAmazon CloudWatch alarmdan menggunakannya untuk mendefinisikanPemeriksaan kondisi Route 53.

Topik

- AWS CloudFormationtemplate untuk menentukan pemeriksaan kondisi Route 53
- Properti templat alarm CloudWatch
- Sifat templat pemeriksaan kondisi 53

AWS CloudFormationtemplate untuk menentukan pemeriksaan kondisi Route 53

Gunakan template berikut untuk menentukan pemeriksaan kondisi Route 53 Anda.

```
Description: |-
  Global endpoints health check that will fail when the average Amazon EventBridge
```

```
  latency is above 30 seconds for a duration of 5 minutes. Note, missing data will
  cause the health check to fail, so if you only send events intermittently, consider
  changing the heath check to use a longer evaluation period or instead treat missing
  data as 'missing' instead of 'breaching'.

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      - Label:
          default: "Global endpoint health check alarm configuration"
        Parameters:
          - HealthCheckName
          - HighLatencyAlarmPeriod
          - MinimumEvaluationPeriod
          - MinimumThreshold
          - TreatMissingDataAs
    ParameterLabels:
      HealthCheckName:
        default: Health check name
      HighLatencyAlarmPeriod:
        default: High latency alarm period
      MinimumEvaluationPeriod:
        default: Minimum evaluation period
      MinimumThreshold:
        default: Minimum threshold
      TreatMissingDataAs:
        default: Treat missing data as

Parameters:
  HealthCheckName:
    Description: Name of the health check
    Type: String
    Default: LatencyFailuresHealthCheck
  HighLatencyAlarmPeriod:
    Description: The period, in seconds, over which the statistic is applied. Valid
 values are 10, 30, 60, and any multiple of 60.
    MinValue: 10
    Type: Number
    Default: 60
  MinimumEvaluationPeriod:
    Description: The number of periods over which data is compared to the specified
 threshold. You must have at least one evaluation period.
    MinValue: 1
    Type: Number
```

```
    Default: 5
  MinimumThreshold:
    Description: The value to compare with the specified statistic.
    Type: Number
    Default: 30000
  TreatMissingDataAs:
    Description: Sets how this alarm is to handle missing data points.
    Type: String
    AllowedValues:
      - breaching
      - notBreaching
      - ignore
      - missing
    Default: breaching

Mappings:
  "InsufficientDataMap":
    "missing":
      "HCConfig": "LastKnownStatus"
    "breaching":
      "HCConfig": "Unhealthy"

Resources:
  HighLatencyAlarm:
      Type: AWS::CloudWatch::Alarm
      Properties:
        AlarmDescription: High Latency in Amazon EventBridge
        MetricName: IngestionToInvocationStartLatency
        Namespace: AWS/Events
        Statistic: Average
        Period: !Ref HighLatencyAlarmPeriod
        EvaluationPeriods: !Ref MinimumEvaluationPeriod
        Threshold: !Ref MinimumThreshold
        ComparisonOperator: GreaterThanThreshold
        TreatMissingData: !Ref TreatMissingDataAs

  LatencyHealthCheck:
      Type: AWS::Route53::HealthCheck
      Properties:
        HealthCheckTags:
          - Key: Name
            Value: !Ref HealthCheckName
        HealthCheckConfig:
          Type: CLOUDWATCH_METRIC
```

```
          AlarmIdentifier:
            Name:
              Ref: HighLatencyAlarm
            Region: !Ref AWS::Region
          InsufficientDataHealthStatus: !FindInMap [InsufficientDataMap, !Ref
 TreatMissingDataAs, HCConfig]

Outputs:
  HealthCheckId:
    Description: The identifier that Amazon Route 53 assigned to the health check when
 you created it.
    Value: !GetAtt LatencyHealthCheck.HealthCheckId
```

ID peristiwa dapat berubah di seluruh panggilan API sehingga menghubungkan peristiwa di seluruh Wilayah mengharuskan Anda memiliki pengenal unik yang tidak berubah. Konsumen juga harus dirancang dengan idempotency dalam pikiran. Dengan begitu, jika Anda mereplikasi acara, atau memutarnya ulang dari arsip, tidak ada efek samping dari peristiwa yang sedang diproses di kedua Wilayah.

Properti templat alarm CloudWatch

> **Note**
>
> Untuk semua**`editable`**bidang, pertimbangkan throughput Anda per detik. Jika Anda hanya mengirim acara sebentar-sebentar, pertimbangkan untuk mengubah pemeriksaan kesehatan untuk menggunakan periode evaluasi yang lebih lama atau sebagai gantinya memperlakukan data yang hilang sebagai`missing`alih-alih`breaching`.

Properti berikut digunakan dalam bentuk CloudWatch bagian alarm dari template:

Metrik	Deskripsi
`AlarmDescription`	Deskripsi alarm. Default: **`High Latency in Amazon EventBridge`**
`MetricName`	Nama metrik yang terkait dengan alarm. Ini diperlukan untuk alarm yang didasarkan pada metrik. Untuk alarm yang didasarkan pada ekspresi

Metrik	Deskripsi
	matematika, Anda menggunakan `Metrics` sebagai gantinya dan tidak dapat menentukan `MetricName` . Default: IngestiontoInvocationStartlatency
`Namespace`	Namespace metrik yang terkait dengan alarm. Ini diperlukan untuk alarm yang didasarkan pada metrik. Untuk alarm yang didasarkan pada ekspresi matematika, Anda tidak dapat menentukan `Namespace` dan menggunakan `Metrics` sebagai gantinya. Default: `AWS/Events`
`Statistic`	Statistik metrik yang terkait dengan alarm, selain persentil. Default: Rata-rata
`Period`	Periode, dalam detik, ketika statistik diterapkan. Ini diperlukan untuk alarm yang didasarkan pada metrik. Nilai yang valid adalah 10, 30, 60, dan kelipatan 60. Default: **`60`**
`EvaluationPeriods`	Jumlah periode yang mana data dibandingkan dengan ambang batas yang ditentukan. Jika Anda mengatur alarm yang mengharuskan sejumlah titik data berturut-turut melanggar untuk memicu alarm, nilai ini menentukan jumlah tersebut. Jika Anda menetapkan alarm "M dari N", nilai ini adalah N, dan `DatapointsToAlarm` adalah M. Default: **`5`**
`Threshold`	Nilai untuk membandingkan dengan statistik yang ditentukan. Default: **`30,000`**

Metrik	Deskripsi
`ComparisonOperator`	Operasi aritmatika yang akan digunakan saat membandingkan ambang batas dan statistik yang ditentukan. Nilai statistik yang ditentukan digunakan sebagai operan pertama. Default: `GreaterThanThreshold`
`TreatMissingData`	Mengatur bagaimana alarm ini menangani titik data yang hilang. Nilai yang benar:`breaching` ,`notBreaching` ,`ignore`, dan`missing` Default: `breaching`

Sifat templat pemeriksaan kondisi 53

> **Note**
>
> Untuk semua**`editable`**bidang, pertimbangkan throughput Anda per detik. Jika Anda hanya mengirim acara sebentar-sebentar, pertimbangkan untuk mengubah pemeriksaan kesehatan untuk menggunakan periode evaluasi yang lebih lama atau sebagai gantinya memperlakukan data yang hilang sebagai`missing`alih-alih`breaching`.

Properti berikut digunakan dalam th Route 53 pemeriksaan kesehatan bagian dari template:

Metrik	Deskripsi
`HealthCheckName`	Nama pemeriksaan kondisi. Default: **`LatencyFailuresHealthCheck`**
`InsufficientDataHealthStatus`	Saat CloudWatch memiliki data yang tidak mencukupi tentang metrik untuk menentukan status alarm, status yang Anda inginkan Amazon Route 53 untuk pemeriksaan kesehatan Nilai yang valid: • `Healthy`: Route 53 menganggap pemeriksaan kondisi sehat.

Metrik	Deskripsi
	• `Unhealthy` : Rute 53 menganggap pemeriksaan kesehatan tidak sehat. • `LastKnownStatus` : Route 53 menggunakan status pemeriksaan kondisi dari saat terakhir CloudWatch memiliki data yang cukup untuk menentukan status alarm. Untuk pemeriksaan kondisi baru yang tidak memiliki status terakhir, status default untuk pemeriksaan kondisi adalah sehat. Default: Tidak Baik **Note** Bidang ini diperbarui berdasarkan input ke`TreatMissingData` bidang. Jika`TreatingMissingData` diatur ke`Missing`, itu akan diperbarui untuk`LastKnownStatus` Jika`TreatingMissingData` diatur ke`Breaching` , itu akan diperbarui untuk`Unhealthy` .

EventBridge Skema Amazon

Skema mendefinisikan struktur peristiwa yang dikirim ke. EventBridge EventBridge menyediakan skema untuk semua acara yang dihasilkan oleh AWS layanan. Anda juga dapat membuat atau mengunggah skema kustom atau mengambil kesimpulan skema secara langsung dari peristiwa pada bus peristiwa. Setelah Anda memiliki skema untuk peristiwa, Anda dapat mengunduh pengikatan kode untuk bahasa pemrograman populer dan mempercepat pengembangan. Anda dapat bekerja dengan pengikatan kode untuk skema dan mengelola skema dari EventBridge konsol, dengan menggunakan API, atau langsung di IDE Anda dengan menggunakan toolkit. AWS Untuk membangun aplikasi nirserver yang menggunakan peristiwa, gunakan AWS Serverless Application Model.

> **Note**
>
> Saat menggunakan fitur transformator input, peristiwa asli disimpulkan dengan penemuan skema, bukan peristiwa yang diubah yang dikirim ke target.

EventBridge mendukung format OpenAPI 3 dan JsonSchema Draft4.

Untuk AWS Toolkit for JetBrains dan AWS Toolkit for VS Code, Anda dapat menelusuri atau mencari skema dan mengunduh binding kode untuk skema langsung di IDE Anda.

Video berikut memberikan gambaran umum tentang skema dan pendaftar skema: Menggunakan Registri Skema

Topik

- Penutupan nilai properti API registri skema
- Menemukan EventBridge skema Amazon
- Pendaftaran EventBridge skema Amazon
- Membuat EventBridge skema Amazon
- EventBridge Binding kode Amazon

Penutupan nilai properti API registri skema

Beberapa nilai properti dari peristiwa yang digunakan untuk membuat registri skema mungkin berisi informasi pelanggan yang sensitif. Untuk melindungi informasi pelanggan, nilai akan ditutupi dengan tanda bintang (*). Karena kita menutupi nilai-nilai ini, EventBridge merekomendasikan untuk tidak membangun aplikasi yang secara eksplisit bergantung pada properti berikut atau nilainya:

- CreateSchema— `Content` Properti `requestParameters` tubuh
- GetDiscoveredSchema— `Events` Properti `requestParameters` tubuh dan `Content` properti `responseElements` tubuh
- SearchSchemas— `keywords` Properti dari `requestParameters`
- UpdateSchema— `Content` Properti dari `requestParameters`

Menemukan EventBridge skema Amazon

EventBridge termasuk skema untuk semua AWS layanan yang menghasilkan acara. Anda dapat menemukan skema ini di EventBridge konsol, atau Anda dapat menemukannya dengan menggunakan tindakan `SearchSchemas`API.

Untuk menemukan skema untuk AWS layanan di konsol EventBridge

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Skema.
3. Pada halaman Skema, pilih AWS registri skema peristiwa.
   <result>

   Halaman pertama skema yang tersedia ditampilkan.
   </result>
4. Untuk menemukan skema, dalam skema AWS acara Penelusuran, masukkan istilah pencarian.

   Pencarian kembali cocok untuk kedua nama dan konten dari skema yang tersedia, dan kemudian menampilkan versi skema yang berisi kecocokan.
5. Buka skema peristiwa dengan memilih nama skema.

Pendaftaran EventBridge skema Amazon

Registri skema adalah kontainer untuk skema. Registri skema mengumpulkan dan mengatur skema sehingga skema Anda berada dalam kelompok logika. Registri skema default adalah:

- Semua skema — Semua skema dari AWS acara, penemuan, dan pendaftar skema khusus.
- AWS registri skema acara - Skema bawaan.
- Registri skema yang ditemukan – Skema yang ditemukan oleh penemuan Skema.

Anda dapat membuat daftar kustom untuk mengatur skema yang Anda buat atau unggah.

Untuk membuat registri kustom

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Skema dan kemudian pilih Buat registri.
3. Pada halaman Detail registri, masukkan Nama.
4. (Opsional) Masukkan deskripsi untuk registri baru Anda.
5. Pilih Buat.

Untuk membuat skema kustom di registri baru Anda, pilih Buat skema kustom. Untuk menambahkan skema ke registri Anda, pilih registri tersebut saat Anda membuat skema baru.

Untuk membuat registri dengan menggunakan API, gunakan `CreateRegistry`. Untuk informasi selengkapnya, lihat Referensi API Registri EventBridge Skema Amazon.

Untuk informasi tentang menggunakan registri EventBridge skema AWS CloudFormation, lihat Referensi Jenis EventSchemas Sumber Daya di AWS CloudFormation.

Membuat EventBridge skema Amazon

Anda membuat skema dengan menggunakan file JSON dengan Spesifikasi OpenAPI maupun spesifikasi JsonSchema Draft4. Anda dapat membuat atau mengunggah skema Anda sendiri EventBridge dengan menggunakan templat atau membuat skema berdasarkan JSON suatu peristiwa. Anda juga dapat menyimpulkan skema dari peristiwa pada bus peristiwa. Untuk membuat skema menggunakan EventBridge Schema Registry API, gunakan tindakan `CreateSchema`API.

Saat Anda memilih antara format OpenAPI 3 dan JsonSchema Draft4, pertimbangkan perbedaan berikut ini:

- Format JsonSchema mendukung kata kunci tambahan yang tidak didukung di OpenAPI, seperti `$schema, additionalItems`.
- Terdapat perbedaan kecil dalam bagaimana kata kunci ditangani, seperti `type` dan `format`.
- OpenAPI tidak mendukung hyperlink JsonSchema Hyper-Schema dalam dokumen JSON.
- Alat untuk OpenAPI cenderung fokus pada build-time, sedangkan alat untuk JsonSchema cenderung fokus pada operasi waktu aktif, seperti alat klien untuk validasi skema.

Sebaiknya gunakan format JsonSchema untuk mengimplementasikan validasi sisi klien sehingga peristiwa yang dikirim agar sesuai dengan skema. EventBridge Anda dapat menggunakan JsonSchema untuk menentukan kontrak untuk dokumen JSON valid, dan kemudian menggunakan validator skema JSON sebelum mengirim peristiwa yang berkaitan.

Setelah Anda memiliki skema baru, Anda dapat mengunduh pengikatan kode untuk membantu membuat aplikasi untuk peristiwa dengan skema tersebut.

Topik

- Membuat skema dengan menggunakan templat
- Edit templat skema secara langsung di konsol tersebut
- Buat skema dari JSON dari peristiwa
- Membuat skema dari peristiwa pada bus peristiwa

Membuat skema dengan menggunakan templat

Anda dapat membuat skema dari template atau dengan mengedit template langsung di EventBridge konsol. Untuk mendapatkan templat, Anda mengunduhnya dari konsol tersebut. Anda dapat

mengedit templat sehingga skema cocok dengan peristiwa Anda. Kemudian unggah templat baru Anda melalui konsol tersebut.

Untuk mengunduh templat skema

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Registri skema.
3. Di bagian Memulai di bawahTemplat skema, pilih Unduh.

Sebagai alternatif, Anda dapat menyalin templat JSON dari contoh kode berikut ini.

```
{
    "openapi": "3.0.0",
    "info": {
      "version": "1.0.0",
      "title": "Event"
    },
    "paths": {},
    "components": {
      "schemas": {
        "Event": {
          "type": "object",
          "properties": {
            "ordinal": {
              "type": "number",
              "format": "int64"
            },
            "name": {
              "type": "string"
            },
            "price": {
              "type": "number",
              "format": "double"
            },
            "address": {
              "type": "string"
            },
            "comments": {
              "type": "array",
              "items": {
                "type": "string"
              }
```

```
            },
            "created_at": {
              "type": "string",
              "format": "date-time"
            }
          }
        }
      }
    }
  }
```

Untuk mengunggah templat skema

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Skema dan kemudian pilih Buat skema.
3. (Opsional) Pilih atau buat registri skema.
4. Di bawah Detail skema, masukkan nama untuk skema Anda.
5. (Opsional) Masukkan deskripsi untuk skema Anda.
6. Untuk Jenis skema, pilih OpenAPI 3.0 maupun JSON Schema Draft 4.
7. Pada tab Buat, di kotak teks, seret file skema ke kotak teks, ataupun tempel sumber skema.
8. Pilih Buat.

Edit templat skema secara langsung di konsol tersebut

Untuk mengedit skema di konsol

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Skema dan kemudian pilih Buat skema.
3. (Opsional) Pilih atau buat registri skema.
4. Di bawah Detail skema, masukkan nama untuk skema Anda.
5. Untuk Jenis skema, pilih OpenAPI 3.0 maupun JSON Schema Draft 4.
6. (Opsional) Masukkan deskripsi untuk skema yang akan dibuat.
7. Pada tab Buat, pilih Muat templat.
8. Dalam kotak teks, edit templat sehingga skema cocok dengan peristiwa Anda.
9. Pilih Buat.

Buat skema dari JSON dari peristiwa

Jika Anda memiliki JSON peristiwa, Anda secara otomatis dapat membuat skema untuk jenis peristiwa.

Untuk membuat skema berdasarkan JSON dari peristiwa

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Skema dan kemudian pilih Buat skema.
3. (Opsional) Pilih atau buat registri skema.
4. Di bawah Detail skema masukkan nama untuk skema Anda.
5. (Opsional) Masukkan deskripsi untuk skema yang Anda buat.
6. Untuk Jenis skema, pilih OpenAPI 3.0.

   Anda tidak dapat menggunakan JsonSchema ketika Anda membuat skema dari JSON peristiwa.
7. Pilih Temukan dari JSON
8. Dalam kotak teks di bawah JSON, tempel atau seret sumber JSON dari peristiwa.

   Misalnya, Anda dapat menempelkan sumber dari AWS Step Functions acara ini untuk eksekusi yang gagal.

```
{
    "version": "0",
    "id": "315c1398-40ff-a850-213b-158f73e60175",
    "detail-type": "Step Functions Execution Status Change",
    "source": "aws.states",
    "account": "012345678912",
    "time": "2019-02-26T19:42:21Z",
    "region": "us-east-1",
    "resources": [
      "arn:aws:states:us-east-1:012345678912:execution:state-machine-
name:execution-name"
    ],
    "detail": {
        "executionArn": "arn:aws:states:us-east-1:012345678912:execution:state-
machine-name:execution-name",
        "stateMachineArn": "arn:aws:states:us-
east-1:012345678912:stateMachine:state-machine",
        "name": "execution-name",
        "status": "FAILED",
```

```
        "startDate": 1551225146847,
        "stopDate": 1551225151881,
        "input": "{}",
        "output": null
    }
}
```

9. Pilih Discover schema (Temukan skema).
10. EventBridge menghasilkan skema OpenAPI untuk acara tersebut. Sebagai contoh, skema berikut ini dihasilkan untuk peristiwa Step Functions sebelumnya.

```
{
  "openapi": "3.0.0",
  "info": {
    "version": "1.0.0",
    "title": "StepFunctionsExecutionStatusChange"
  },
  "paths": {},
  "components": {
    "schemas": {
      "AWSEvent": {
        "type": "object",
        "required": ["detail-type", "resources", "detail", "id", "source", "time",
 "region", "version", "account"],
        "x-amazon-events-detail-type": "Step Functions Execution Status Change",
        "x-amazon-events-source": "aws.states",
        "properties": {
          "detail": {
            "$ref": "#/components/schemas/StepFunctionsExecutionStatusChange"
          },
          "account": {
            "type": "string"
          },
          "detail-type": {
            "type": "string"
          },
          "id": {
            "type": "string"
          },
          "region": {
            "type": "string"
          },
          "resources": {
```

```
            "type": "array",
            "items": {
              "type": "string"
            }
          },
          "source": {
            "type": "string"
          },
          "time": {
            "type": "string",
            "format": "date-time"
          },
          "version": {
            "type": "string"
          }
        }
      },
      "StepFunctionsExecutionStatusChange": {
        "type": "object",
        "required": ["output", "input", "executionArn", "name", "stateMachineArn",
"startDate", "stopDate", "status"],
        "properties": {
          "executionArn": {
            "type": "string"
          },
          "input": {
            "type": "string"
          },
          "name": {
            "type": "string"
          },
          "output": {},
          "startDate": {
            "type": "integer",
            "format": "int64"
          },
          "stateMachineArn": {
            "type": "string"
          },
          "status": {
            "type": "string"
          },
          "stopDate": {
            "type": "integer",
```

```
            "format": "int64"
          }
        }
      }
    }
  }
}
```

11. Setelah skema telah dihasilkan, pilih Buat.

Membuat skema dari peristiwa pada bus peristiwa

EventBridge dapat menyimpulkan skema dengan menemukan peristiwa. Untuk menyimpulkan skema, Anda mengaktifkan penemuan acara di bus acara dan setiap skema unik ditambahkan ke registri skema, termasuk skema untuk acara lintas akun. Skema yang ditemukan oleh EventBridge muncul di registri skema Ditemukan di halaman Skema.

Jika isi acara di bus acara berubah, EventBridge buat versi baru dari EventBridge skema terkait.

> **Note**
>
> Mengaktifkan penemuan peristiwa pada bus peristiwa dapat dikenakan biaya. Lima juta peristiwa pertama yang diproses setiap bulannya adalah gratis.

> **Note**
>
> EventBridge menyimpulkan skema dari peristiwa lintas akun secara default tetapi Anda dapat menonaktifkannya dengan memperbarui properti. `cross-account` Untuk informasi selengkapnya, lihat Penemu di Refernce API Registri EventBridge Skema.

Untuk mengaktifkan penemuan skema pada bus peristiwa

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Bus peristiwa.
3. Lakukan salah satu dari berikut ini:
   - Untuk mengaktifkan penemuan pada Bus peristiwa default, pilih Mulai penemuan.

- Untuk mengaktifkan penemuan pada Bus peristiwa kustom, pilih tombol radio untuk bus peristiwa kustom dan kemudian pilih Mulai penemuan.

EventBridge Binding kode Amazon

Anda dapat menghasilkan binding kode untuk skema acara untuk mempercepat pengembangan di Golang, Java, Python, dan. TypeScript Pengikatan kode tersedia untuk peristiwa layanan AWS , skema yang Anda buat, dan untuk skema yang Anda hasilkanberdasarkan peristiwa pada bus peristiwa. Anda dapat membuat binding kode untuk skema dengan menggunakan EventBridge konsol, EventBridge Schema Registry API, atau di IDE Anda dengan toolkit. AWS

Untuk menghasilkan binding kode dari skema EventBridge

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Skema.
3. Temukan skema pengikatan kode yang Anda inginkan, baik dengan penelusuran registri skema, atau dengan mencari skema.
4. Pilih nama skema..
5. Pada halaman Detail skema, di bagian Versi, pilih Unduh pengikatan kode.
6. Pada halaman Unduh pengikatan kode, pilih bahasa pengikatan kode yang ingin Anda unduh.
7. Pilih Unduh.

   Mungkin perlu beberapa detik agar unduhan Anda dimulai. File yang diunduh adalah file zip dari pengikatan kode untuk bahasa yang Anda pilih.

Layanan dan alat EventBridge terkait Amazon

Amazon EventBridge bekerja dengan AWS layanan dan alat lain untuk memproses peristiwa atau memanggil sumber daya sebagai target aturan. Untuk informasi selengkapnya tentang EventBridge integrasi dengan AWS layanan dan alat lain, lihat berikut ini:

Topik

- Menggunakan Amazon EventBridge dengan Titik Akhir VPC Antarmuka
- Integrasi Amazon EventBridge dengan AWS X-Ray
- Menggunakan EventBridge dengan Kit Uji Aplikasi AWS Terpadu
- Termasuk EventBridge sumber daya Amazon dalam AWS CloudFormation tumpukan

Menggunakan Amazon EventBridge dengan Titik Akhir VPC Antarmuka

Jika Anda menggunakan Amazon Virtual Private Cloud (Amazon VPC) untuk meng-host sumber daya AWS Anda, Anda dapat membuat koneksi privat antara VPC dan EventBridge. Sumber daya Anda di VPC Anda dapat menggunakan koneksi ini untuk berkomunikasi. EventBridge

Dengan VPC, Anda memiliki kendali terhadap pengaturan jaringan, seperti rentang alamat IP, subnet, tabel rute, dan gateway jaringan. Untuk menghubungkan VPC Anda EventBridge, Anda menentukan titik akhir VPC antarmuka untuk. EventBridge Endpoint menyediakan konektivitas yang andal dan dapat diskalakan EventBridge tanpa memerlukan gateway internet, instance terjemahan alamat jaringan (NAT), atau koneksi VPN. Untuk informasi selengkapnya, lihat Apa yang dimaksud dengan Amazon VPC dalam Panduan Pengguna Amazon VPC.

Endpoint VPC antarmuka didukung oleh AWS PrivateLink, yang memungkinkan komunikasi pribadi antara AWS layanan menggunakan antarmuka jaringan elastis dengan alamat IP pribadi. Untuk informasi selengkapnya, lihat AWS PrivateLink dan titik akhir VPC.

Saat Anda menggunakan titik akhir VPC antarmuka pribadi, peristiwa khusus yang dikirim VPC Anda untuk menggunakan titik akhir tersebut. EventBridge EventBridge kemudian mengirimkan peristiwa tersebut ke AWS layanan lain berdasarkan aturan dan target yang telah Anda konfigurasikan. Setelah peristiwa dikirim ke layanan lain Anda dapat menerimanya melalui titik akhir publik atau VPC endpoint untuk layanan tersebut. Misalnya, jika Anda membuat aturan untuk mengirim peristiwa ke antrean Amazon SQS, Anda dapat mengonfigurasi antarmuka VPC endpoint untuk Amazon SQS agar dapat menerima pesan dari antrean di VPC Anda tanpa menggunakan titik akhir publik.

Ketersediaan

EventBridge saat ini mendukung titik akhir VPC di Wilayah berikut:

- AS Timur (Ohio)
- AS Timur (Virginia Utara)
- AS Barat (California Utara)
- AS Barat (Oregon)
- Africa (Cape Town)
- Asia Pasifik (Mumbai)
- Asia Pasifik (Hyderabad)

- Asia Pasifik (Hong Kong)
- Asia Pasifik (Singapura)
- Asia Pasifik (Sydney)
- Asia Pasifik (Jakarta)
- Asia Pasifik (Melbourne)
- Asia Pasifik (Tokyo)
- Asia Pasifik (Seoul)
- Asia Pasifik (Osaka)
- Kanada (Pusat)
- Kanada Barat (Calgary)
- Tiongkok (Beijing)
- Tiongkok (Ningxia)
- Eropa (Frankfurt)
- Eropa (Zürich)
- Eropa (Irlandia)
- Eropa (London)
- Eropa (Milan)
- Eropa (Spanyol)
- Eropa (Paris)
- Eropa (Stockholm)
- Timur Tengah (UEA)
- Timur Tengah (Bahrain)
- Amerika Selatan (Sao Paulo)
- Israel (Tel Aviv)
- AWS GovCloud (AS-Barat)
- AWS GovCloud (AS-Timur)

Membuat Endpoint VPC untuk EventBridge

Untuk menggunakan EventBridge VPC Anda, buat antarmuka VPC endpoint untuk dan pilih com.amazonaws. EventBridge ***Region*** .events sebagai nama layanan. Untuk informasi

selengkapnya, silakan lihat Membuat sebuah Titik Akhir Antarmuka dalam Panduan Pengguna Amazon VPC.

EventBridge Spesifikasi pipa

Dukungan EventBridge Pipa Penuh untuk titik akhir VPC Antarmuka tidak tersedia. Untuk menggunakan sumber-sumber berikut dalam VPC dengan EventBridge Pipa, lihat berikut ini:

- Konfigurasi jaringan MSK Amazon
- Konfigurasi jaringan Apache Kafka yang dikelola sendiri
- Konfigurasi jaringan Amazon MQ

Integrasi Amazon EventBridge dengan AWS X-Ray

Anda dapat menggunakan AWS X-Ray untuk melacak peristiwa yang melewati EventBridge. EventBridge melewati header jejak asli untuk target sehingga layanan target dapat melacak, menganalisis, dan debug.

EventBridge dapat melewati header jejak untuk acara hanya jika peristiwa berasal dari `PutEvents` permintaan yang melewati konteks jejak. X-Ray tidak melacak peristiwa yang berasal dari mitra pihak ke tiga, peristiwa terjadwal, atau AWS layanan, dan sumber peristiwa ini tidak muncul di peta layanan X-Ray.

X-Ray memvalidasi header jejak, dan header jejak yang tidak valid akan dijatuhkan. Namun, peristiwa tersebut masih diproses.

> **Important**
>
> Header jejak tidak tersedia pada peristiwa yang dikirim ke target permintaan.
>
> - Jika Anda memiliki arsip peristiwa, header jejak tidak tersedia pada peristiwa yang diarsipkan. Jika Anda memutar ulang peristiwa yang diarsipkan, header jejak tidak disertakan.
> - Jika Anda memiliki antrean surat mati (DLQ), header jejak termasuk dalam `SendMessage` permintaan yang mengirimkan peristiwa ke DLQ. Jika Anda mengambil peristiwa (pesan) dari DLQ menggunakan `ReceiveMessage`, header jejak yang terkait dengan peristiwa, disertakan pada atribut pesan Amazon SQS, tetapi tidak termasuk dalam pesan peristiwa.

Untuk informasi tentang cara simpul peristiwa EventBridge menghubungkan sumber dan layanan target, lihat Melihat sumber dan target dalam peta layanan X-Ray dalam AWS X-Ray Panduan Developer.

Anda dapat melalui informasi header jejak berikut melalui EventBridge:

- Header HTTP default — X-Ray SDK secara otomatis mengisi header jejak sebagai `X-Amzn-Trace-Id` HTTP header untuk semua target permintaan. Untuk mempelajari selengkapnya tentang Header HTTP default, lihat Header jejak dalam AWS X-Ray Panduan Developer..
- **`TraceHeader`** atribut sistem — `TraceHeader` adalah Atribut PutEventsRequestEntry yang dicadangkan oleh EventBridge untuk membawa header jejak X-Ray ke target. Jika Anda juga

menggunakan `PutEventsRequestEntry`, `PutEventsRequestEntry` menimpa header jejak HTTP.

> **Note**
>
> Jejak header tidak dihitung ke arah `PutEventsRequestEntry` ukuran peristiwa. Untuk informasi selengkapnya, lihat Menghitung ukuran entri EventBridge PutEvents acara Amazon.

Video berikut menunjukkan penggunaan X-Ray dan EventBridge bersama-sama: MenggunakanAWS X-Rayuntuk penelusuran

Menggunakan EventBridge dengan Kit Uji Aplikasi AWS Terpadu

Saat Anda membuat aplikasi yang terdiri dari layanan tanpa server seperti Lambda,, atau EventBridge Step Functions, banyak komponen arsitektur Anda tidak dapat digunakan ke desktop Anda, tetapi hanya ada di cloud. AWS Berbeda dengan bekerja dengan aplikasi yang digunakan secara lokal, jenis aplikasi ini mendapat manfaat dari strategi berbasis cloud untuk melakukan pengujian otomatis. AWS Integrated Application Test Kit (AWS IATK) membantu Anda menerapkan beberapa strategi ini untuk aplikasi Anda.

AWS IATK adalah pustaka perangkat lunak yang membantu Anda menulis pengujian otomatis untuk aplikasi berbasis cloud.

EventBridge Integrasi dengan AWS IATK

Anda dapat menggunakan bus EventBridge acara dan acara dengan AWS IATK untuk mengimplementasikan pengujian otomatis Anda, termasuk:

Menerapkan test harness

Untuk menulis tes integrasi untuk arsitektur berbasis peristiwa, buat batasan logis dengan memecah aplikasi Anda menjadi subsistem. Salah satu teknik yang berguna untuk menguji subsistem adalah membuat test harness; yaitu, sumber daya yang Anda buat khusus untuk menguji subsistem.

Misalnya, tes integrasi dapat memulai proses subsistem dengan meneruskan peristiwa pengujian input ke sana. AWS IATK dapat membuat test harness untuk Anda yang mendengarkan EventBridge acara keluaran. (Di bawah tenda, harness terdiri dari EventBridge aturan yang meneruskan peristiwa keluaran ke Amazon SQS.) Tes integrasi Anda kemudian menanyakan harness pengujian untuk memeriksa output dan menentukan apakah tes lulus atau gagal.

Menghasilkan acara tiruan

AWS IATK menyediakan kemampuan bagi Anda untuk menghasilkan peristiwa tiruan dari skema yang disimpan dalam registri skema. EventBridge Ini memungkinkan Anda untuk membuat peristiwa tiruan dan memanggil konsumen mana pun (seperti fungsi Lambda atau mesin status Step Functions) dengan peristiwa yang dihasilkan.

Untuk informasi selengkapnya, lihat Ikhtisar Kit Uji Aplikasi AWS Terpadu di GitHub.

Termasuk EventBridge sumber daya Amazon dalam AWS CloudFormation tumpukan

AWS CloudFormationmemungkinkan Anda mengonfigurasi dan mengelola AWS sumber daya Anda di seluruh akun dan wilayah secara terpusat dan berulang dengan memperlakukan infrastruktur sebagai kode. CloudFormation melakukan ini dengan membiarkan Anda membuat template, yang menentukan sumber daya yang ingin Anda sediakan dan kelola. Sumber daya ini dapat mencakup EventBridge artefak seperti bus dan aturan acara, pipa, skema, dan jadwal, antara lain. Gunakan sumber daya ini untuk memasukkan EventBridge fungsionalitas dalam tumpukan teknologi yang Anda sediakan dan kelola. CloudFormation

EventBridge Sumber daya Amazon tersedia di AWS CloudFormation

EventBridge menyediakan sumber daya untuk digunakan dalam CloudFormation template di ruang nama sumber daya berikut:

- AWS::Events

  Contoh template meliputi:

  - Membuat tujuan API untuk PagerDuty
  - Buat sebuah tujuan API untuk Slack
  - Buat koneksi dengan parameter ApiKey otorisasi

  - Buat koneksi dengan parameter otorisasi OAuth
  - Buat titik akhir global dengan replikasi acara
  - Tolak kebijakan menggunakan beberapa prinsip dan tindakan
  - Berikan izin kepada organisasi menggunakan bus acara khusus
  - Buat aturan Lintas wilayah
  - Buat aturan yang mencakup antrian huruf mati untuk target
  - Secara teratur memanggil fungsi Lambda
  - Memanggil fungsi Lambda sebagai respons terhadap suatu peristiwa
  - Beri tahu topik sebagai tanggapan terhadap log nntry
- AWS::EventSkema
- AWS::Pipes

  Contoh template meliputi:

  - Buat pipa dengan filter acara
- AWS::Scheduler

Menghasilkan definisi EventBridge sumber daya Amazon untuk AWS CloudFormation templat

Sebagai bantuan untuk membantu Anda mulai mengembangkan CloudFormation template, EventBridge konsol memungkinkan Anda membuat CloudFormation template dari bus acara, aturan, dan pipa yang ada di akun Anda.

- ???
- ???
- ???

Mengelola acara AWS CloudFormation tumpukan menggunakan EventBridge

Selain menyertakan EventBridge sumber daya di CloudFormation tumpukan Anda, Anda dapat menggunakannya EventBridge untuk mengelola peristiwa yang dihasilkan oleh CloudFormation tumpukan itu sendiri. CloudFormation mengirimkan peristiwa ke EventBridge setiap kali operasi

membuat, memperbarui, menghapus, atau deteksi drift dilakukan pada tumpukan. CloudFormation juga mengirimkan peristiwa ke EventBridge perubahan status ke set tumpukan dan instance set tumpukan. Anda dapat menggunakan EventBridge aturan untuk merutekan peristiwa ke target yang ditentukan.

Untuk informasi selengkapnya, lihat Mengelola CloudFormation peristiwa menggunakan EventBridge Panduan AWS CloudFormation Pengguna.

Tutorial Amazon EventBridge

EventBridge terintegrasi dengan sejumlahAWSlayanan dan mitra SaaS. Tutorial ini dirancang untuk membantu Anda terbiasa dengan dasar-dasar EventBridge dan bagaimana hal itu dapat menjadi bagian dari arsitektur tanpa server Anda.

Tutorial:

- Amazon EventBridge memulai tutorial
- Amazon EventBridge tutorial untuk mengintegrasikan dengan lainnyaAWSjasa
- Amazon EventBridge tutorial untuk mengintegrasikan dengan penyedia SaaS

Amazon EventBridge memulai tutorial

Tutorial berikut ini membantu Anda menjelajahi fitur EventBridge dan cara menggunakannya.

Tutorial:

- Arsip dan putar ulang EventBridge peristiwa Amazon
- Membuat aplikasi EventBridge sampel Amazon
- Tutorial: Mengunduh pengikatan kode untuk acara menggunakan EventBridge registri skema
- Tutorial: Gunakan transformator input untuk menyesuaikan apa yang EventBridge lolos ke target peristiwa

Arsip dan putar ulang EventBridge peristiwa Amazon

Anda dapat menggunakan EventBridge untuk merutekan peristiwa ke AWS Lambdafungsi tertentu menggunakan aturan.

Di tutorial ini, Anda akan membuat fungsi untuk digunakan sebagai target untuk EventBridge aturan menggunakan konsol Lambda. Kemudian, Anda akan membuat arsip dan aturan yang akan mengarsipkan peristiwa pengujian menggunakan EventBridge konsol. Setelah ada peristiwa di arsip itu, Anda akan memutar ulang mereka.

Langkah:

- Langkah 1: Membuat fungsi Lambda
- Langkah 2: Buat Arsip
- Langkah 3: Buat aturan
- Langkah 4: Kirim peristiwa uji
- Langkah 5: Putar ulang peristiwa
- Langkah 6: Bersihkan sumber daya Anda

Langkah 1: Membuat fungsi Lambda

Pertama, buat. fungsi Lambda untuk mencatat peristiwa.

Untuk membuat fungsi Lambda:

1. Buka konsol AWS Lambda tersebut di https://console.aws.amazon.com/lambda/.
2. Pilih Buat fungsi.
3. Pilih Tulis dari scratch.
4. Masukkan nama dan deskripsi untuk fungsi Lambda. Misalnya, beri nama fungsi tersebut `LogScheduledEvent`.
5. Tinggalkan sisa pilihan sebagai default dan pilih Buat fungsi.
6. Pada tab Kode dari halaman fungsi, klik dua kali index.js.
7. Ganti JavaScript kode yang ada dengan kode berikut:

```
'use strict';

exports.handler = (event, context, callback) => {
```

```
    console.log('LogScheduledEvent');
    console.log('Received event:', JSON.stringify(event, null, 2));
    callback(null, 'Finished');
};
```

8. Pilih Deploy.

Langkah 2: Buat Arsip

Selanjutnya, buat arsip yang akan menampung semua peristiwa pengujian.

Untuk membuat arsip

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Arsip.
3. Pilih Buat arsip.
4. Masukkan nama dan deskripsi untuk arsip. Misalnya, beri nama arsip`ArchiveTest`.
5. Tinggalkan opsi lainnya sebagai default dan pilih Berikutnya.
6. Pilih Buat arsip.

Langkah 3: Buat aturan

Buat aturan untuk mengarsipkan peristiwa yang dikirim ke bus acara.

Untuk membuat tabel

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan. Misalnya, beri nama aturannya`ARTestRule`.

   Aturan tidak boleh memiliki nama yang sama dengan aturan lain di Wilayah yang sama dan di bus kejadian yang sama.
5. Untuk bus Acara, pilih bus kejadian yang ingin Anda kaitkan dengan aturan ini. Jika Anda ingin aturan ini cocok dengan peristiwa yang berasal dari akun Anda, pilih default. Saat layanan AWS di akun Anda menghasilkan kejadian, layanan tersebut akan selalu masuk ke bus kejadian default akun Anda.

6. Untuk jenis Aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.
8. Untuk Sumber acara, pilih Lainnya.
9. Untuk pola Event, masukkan yang berikut ini:

```
{
  "detail-type": [
    "customerCreated"
  ]
}
```

10. Pilih Selanjutnya.
11. Untuk jenis Target, pilih AWSlayanan.
12. Untuk Pilih target, pilih fungsi Lambda dari daftar drop-down.
13. Untuk Fungsi, pilih fungsi Lambda yang Anda buat. di bagian Fungsi Lambda. Dalam contoh ini, pilih`LogScheduledEvent`.
14. Pilih Selanjutnya.
15. Pilih Selanjutnya.
16. Tinjau detail aturan dan pilih Buat aturan.

Langkah 4: Kirim peristiwa uji

Sekarang setelah Anda menyiapkan arsip dan aturannya, kami akan mengirimkan peristiwa pengujian untuk memastikan arsip berfungsi dengan benar.

> **Note**
>
> Proses pembuatan acara dapat memakan waktu lama untuk sampai ke arsip.

Untuk mengirim peristiwa pengujian (konsol)

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Bus peristiwa.
3. Di ubin bus acara default, pilih Tindakan, Kirim acara.

4. Masukkan sumber peristiwa. Sebagai contoh, `TestEvent`.
5. Untuk jenis Detail, masukkan`customerCreated`.
6. Untuk detail Acara, masukkan`{}`.
7. Pilih Kirim.

Langkah 5: Putar ulang peristiwa

Setelah peristiwa pengujian berada di arsip, Anda dapat memutar ulang mereka.

Untuk memutar ulang peristiwa yang diarsipkan (konsol)

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Putar ulang.
3. Pilih Mulai replay baru.
4. Masukkan nama dan deskripsi untuk pemutaran ulang. Misalnya, beri nama replay`ReplayTest`.
5. Untuk Sumber, pilih arsip yang Anda buat di Langkah 2: Buat arsip bagian.
6. Untuk kerangka waktu Replay, lakukan hal berikut.
   a. Untuk Waktu mulai, pilih tanggal Anda mengirim peristiwa pengujian dan waktu sebelum Anda mengirimnya. Misalnya, `2021/08/11` dan `08:00:00`.
   b. Untuk Waktu akhir, pilih tanggal dan waktu saat ini. Misalnya, `2021/08/11` dan `09:15:00`.
7. Pilih Mulai Putar Ulang.

Langkah 6: Bersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankan itu. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus fungsi Lambda

1. Buka halaman Fungsi di konsol Lambda.
2. Pilih fungsi yang Anda buat.
3. Pilih Tindakan, Hapus.
4. Pilih Hapus.

Untuk menghapus EventBridge arsip

1. Buka halaman Arsip EventBridge konsol.
2. Pilih arsip yang Anda buat.
3. Pilih Delete (Hapus).
4. Masukkan nama arsip dan pilih Hapus.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Membuat aplikasi EventBridge sampel Amazon

Anda dapat menggunakannya EventBridge untuk merutekan peristiwa ke fungsi Lambda tertentu menggunakan aturan.

Dalam tutorial ini, Anda akan menggunakanAWS CLI, Node.js, dan kode di GitHubrepo untuk membuat yang berikut:

- AWS LambdaFungsi yang menghasilkan acara untuk transaksi ATM bank.
- Tiga fungsi Lambda digunakan sebagai target EventBridge aturan.
- dan aturan yang merutekan peristiwa yang dibuat ke fungsi hilir yang benar berdasarkan pola acara.

Contoh ini menggunakanAWS SAM template untuk menentukan EventBridge aturan. Untuk mempelajari lebih lanjut tentang menggunakanAWS SAM template dengan EventBridge lihat???.

Di repo, subdirektori ATMProducer berisi`handler.js`, yang mewakili acara produksi layanan ATM. Kode ini adalah handler Lambda yang ditulis dalam Node.js, dan menerbitkan event ke EventBridge melalui AWSSDK menggunakan baris JavaScript kode ini.

```
const result = await eventbridge.putEvents(params).promise()
```

Direktori ini juga berisi`events.js`, daftar beberapa transaksi uji dalam array Entries. Sebuah acara tunggal didefinisikan JavaScript sebagai berikut:

```
{
  // Event envelope fields
  Source: 'custom.myATMapp',
  EventBusName: 'default',
  DetailType: 'transaction',
  Time: new Date(),

  // Main event body
  Detail: JSON.stringify({
    action: 'withdrawal',
    location: 'MA-BOS-01',
    amount: 300,
    result: 'approved',
    transactionId: '123456',
```

```
    cardPresent: true,
    partnerBank: 'Example Bank',
    remainingFunds: 722.34
  })
}
```

Bagian Detail dari acara menentukan atribut transaksi. Ini termasuk lokasi ATM, jumlah, bank mitra, dan hasil transaksi.

`handler.js`File dalam subdirektori ATMConsumer berisi tiga fungsi:

```
exports.case1Handler = async (event) => {
  console.log('--- Approved transactions ---')
  console.log(JSON.stringify(event, null, 2))
}

exports.case2Handler = async (event) => {
  console.log('--- NY location transactions ---')
  console.log(JSON.stringify(event, null, 2))
}

exports.case3Handler = async (event) => {
  console.log('--- Unapproved transactions ---')
  console.log(JSON.stringify(event, null, 2))
}
```

Setiap fungsi menerima peristiwa transaksi, yang dicatat melalui`console.log` pernyataan ke Amazon CloudWatch Logs. Fungsi konsumen beroperasi secara independen dari produsen dan tidak menyadari sumber peristiwa.

Logika routing terkandung dalam EventBridge aturan yang digunakan olehAWS SAM template aplikasi. Aturan mengevaluasi aliran peristiwa yang masuk, dan merutekan peristiwa yang cocok dengan fungsi Lambda target.

Aturan menggunakan pola peristiwa yang objek JSON dengan struktur yang sama dengan peristiwa mereka cocokkan. Berikut adalah pola acara untuk salah satu aturan.

```
{
  "detail-type": ["transaction"],
  "source": ["custom.myATMapp"],
  "detail": {
    "location": [{
```

```
      "prefix": "NY-"
    }]
  }
}
```

Langkah:

- Prasyarat
- Langkah 1: Buat aplikasi
- Langkah 2: Jalankan aplikasi
- Langkah 3: Periksa log dan verifikasi karya aplikasi
- Langkah 4: Bersihkan sumber daya Anda

Prasyarat

Untuk menyelesaikan tutorial ini, Anda memerlukan sumber daya berikut:

- Akun AWS. BuatAWS akun jika Anda belum memilikinya.
- AWS CLIdipasang. Untuk menginstal AWS CLI, lihat Menginstal, memperbarui, dan menghapus instalasi AWS CLI versi 2.
- Node.js 12.x diinstal. Untuk menginstal Node.js, lihat Unduhan.

Langkah 1: Buat aplikasi

Untuk menyiapkan aplikasi contoh, Anda akan menggunakanAWS CLI dan Git untuk membuatAWS sumber daya yang Anda perlukan.

Untuk membuat aplikasi

1. Masuk keAWS.
2. Instal Git dan instalAWS Serverless Application Model CLI di komputer lokal Anda.
3. Buat direktori baru, dan kemudian arahkan ke direktori itu di terminal.
4. Di baris perintah, masukkan`git clone https://github.com/aws-samples/amazon-eventbridge-producer-consumer-example`.
5. Di baris perintah jalankan perintah berikut:

   ```
   cd ./amazon-eventbridge-producer-consumer-example
   ```

```
sam deploy --guided
```

6. Di terminal, lakukan hal berikut:

   a. Untuk**`Stack Name`**, masukkan nama untuk stack. Misalnya, beri nama stack`Test`.

   b. Untuk**`AWS Region`**, masukkan Wilayah. Sebagai contoh, `us-west-2`.

   c. Untuk **`Confirm changes before deploy`**, masukkan `Y`.

   d. Untuk**`Allow SAM CLI IAM role creation`**, masukkan`Y`

   e. Untuk**`Save arguments to configuration file`**, masukkan`Y`

   f. Untuk **`SAM configuration file`**, masukkan `samconfig.toml`.

   g. Untuk **`SAM configuration environment`**, masukkan `default`.

Langkah 2: Jalankan aplikasi

Sekarang setelah Anda menyiapkan sumber daya, Anda akan menggunakan konsol untuk menguji fungsi.

Untuk menjalankan aplikasi

1. Buka konsol Lambda di Wilayah yang sama tempat Anda menerapkanAWS SAM aplikasi.
2. Ada empat fungsi Lambda dengan awalan atm-demo. Pilih atmProducerFnfungsinya, lalu pilih Actions, Test.
3. Masukkan`Test` untuk Nama.
4. Pilih Uji.

Langkah 3: Periksa log dan verifikasi karya aplikasi

Sekarang setelah Anda menjalankan aplikasi, Anda akan menggunakan konsol untuk memeriksa CloudWatch Log.

Untuk memeriksa log

1. Buka CloudWatch konsol di Wilayah yang sama di mana Anda menjalankanAWS SAM aplikasi.
2. Pilih Log, lalu pilih Log grup.
3. Pilih grup log yang berisi atmConsumerCase1. Anda melihat dua aliran mewakili dua transaksi yang disetujui oleh ATM. Pilih stream log untuk melihat output.

4. Arahkan kembali ke daftar grup log, lalu pilih grup log yang berisi atmConsumerCase2. Anda akan melihat dua aliran yang mewakili dua transaksi yang cocok dengan filter lokasi New York.
5. Arahkan kembali ke daftar grup log, dan pilih grup log yang berisi atmConsumerCase3. Buka streaming untuk melihat transaksi yang ditolak.

Langkah 4: Bersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankannya. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih kebijakan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Untuk menghapus fungsi Lambda

1. Buka halaman Fungsi di konsol Lambda.
2. Pilih fungsi yang Anda buat.
3. Pilih Tindakan, Hapus.
4. Pilih Hapus.

Untuk menghapus grup CloudWatch log log

1. Buka konsol Cloudwatch.
2. Pilih Log, Log grup.
3. Pilih grup log yang dibuat dalam tutorial ini.
4. Pilih Actions (Tindakan), Delete log group(s) (Hapus grup log).
5. Pilih Hapus.

Tutorial: Mengunduh pengikatan kode untuk acara menggunakan EventBridge registri skema

Anda dapat menghasilkanpengikatan kodeuntukSkema peristiwauntuk mempercepat pengembangan Golang, Java, Python, dan TypeScript. Anda bisa mendapatkan pengikatan kode untuk yang adaAWS layanan, skema yang Anda buat, dan skema yang Anda hasilkan berdasarkan peristiwa pada bus acara. Anda dapat menghasilkan pengikatan kode untuk skema menggunakan salah satu dari berikut:

- Konsol EventBridge
- API registri skema EventBridge
- IDE Anda dengan AWS kit alat

Dalam tutorial ini Anda menghasilkan dan mengunduh pengikatan kode dari sebuah EventBridge skema untuk peristiwaAWSlayanan.

Untuk menghasilkan pengikatan kode dari sebuah EventBridge skema

1. Buka Amazon EventBridge konsol dihttps://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Skema.
3. Pilih tab AWS peristiwa registri.
4. Menemukan skema untuk AWS layanan yang Anda ingin pengikatan kode, baik dengan browsing melalui registri skema, atau dengan Telisik skema.
5. Pilih nama skema.
6. PadaDetail skema halaman, di bagian Versi, pilih Mengunduh pengikatan kode.
7. Pada halaman Mengunduh pengikatan kode, pilih bahasa pengikatan kode yang ingin Anda unduh.
8. Pilih Unduh.

   Mungkin perlu beberapa detik agar unduhan Anda dimulai. File unduhan akan menjadi file .zip (zip) pengikatan kode untuk bahasa yang Anda pilih.
9. Unzip file yang diunduh dan tambahkan ke proyek Anda.

   Paket unduhan berisi file README yang menjelaskan cara mengkonfigurasi dependensi paket dalam berbagai kerangka kerja.

Gunakan pengikatan kode ini dalam kode Anda sendiri untuk membantu dengan cepat membangun aplikasi menggunakan ini EventBridge peristiwa.

Tutorial: Gunakan transformator input untuk menyesuaikan apa yang EventBridge lolos ke target peristiwa

Anda dapat menggunakan transformator Input EventBridge untuk menyesuaikan teks dari suatu peristiwa sebelum Anda mengirimkannya ke target aturan.

Untuk melakukan ini, Anda menentukan jalur JSON dari acara tersebut dan menetapkan output mereka ke variabel yang berbeda. Kemudian Anda dapat menggunakan variabel-variabel dalam templat masukan. Karakter < and > tidak dapat lepas. Untuk informasi selengkapnya, lihat Transformasi EventBridge masukan Amazon

> **Note**
>
> Jika Anda menentukan variabel untuk mencocokkan jalur JSON yang tidak ada dalam acara, variabel tersebut tidak dibuat dan tidak muncul dalam output.

Dalam tutorial ini, Anda membuat aturan yang cocok dengan peristiwa`detail-type: "customerCreated"`. Transformator masukan memetakan`type` variabel ke jalur JSON $.detail-type dari acara tersebut. Kemudian EventBridge menempatkan variabel ke dalam template masukan “Acara ini adalah<type>.” Hasilnya adalah pesan Amazon SNS berikut.

```
"This event was of customerCreated type."
```

Langkah:

- Langkah 1: Buat Topik Amazon SNS
- Langkah 2: Buat langganan Amazon SNS
- Langkah 3: Buat aturan
- Langkah 4: Kirim peristiwa uji
- Langkah 5: Mengonfirmasi keberhasilan
- Langkah 6: Bersihkan sumber daya Anda

Langkah 1: Buat Topik Amazon SNS

Buat topik untuk menerima acara dari EventBridge.

Cara membuat topik

1. Buka konsol Amazon SNS di https://console.aws.amazon.com/sns/v3/home.
2. Di panel navigasi, pilih Topik.
3. Pilih Buat topik.
4. Untuk Tipe, pilih Standar.
5. Masukkan**`eventbridge-IT-test`** sebagai nama topik.
6. Pilih Buat topik.

Langkah 2: Buat langganan Amazon SNS

Buat langganan untuk mendapatkan email dengan informasi yang ditransformasikan.

Untuk membuat langganan

1. Buka konsol Amazon SNS di https://console.aws.amazon.com/sns/v3/home.
2. Di panel navigasi, pilih Subscriptions (Langganan).
3. Pilih Create subscription (Buat langganan).
4. Untuk Topik ARN, pilih topik yang Anda buat di langkah 1. Untuk tutorial ini, pilih EventBridge-IT-Test.
5. Untuk Protokol, pilih Email.
6. Untuk Titik Akhir, masukkan alamat email Anda.
7. Pilih Buat langganan.
8. Konfirmasikan langganan dengan memilih Konfirmasi langganan di email yang Anda terima dariAWS notifikasi.

Langkah 3: Buat aturan

Buat aturan untuk menggunakan trafo input untuk menyesuaikan informasi keadaan instans yang masuk ke target.

Untuk membuat tabel

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.

3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan. Misalnya, beri nama aturan`ARTestRule`
5. Untuk Pilihan bus kejadian, pilih bus kejadian yang ingin Anda kaitkan dengan aturan ini. Jika Anda ingin aturan ini cocok dengan kejadian yang berasal dari akun Anda, pilih default. Saat layanan AWS di akun Anda menghasilkan kejadian, layanan tersebut akan selalu masuk ke bus kejadian default akun Anda.
6. Untuk jenis Aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.
8. Untuk Sumber acara, pilih Lainnya.
9. Untuk pola Event, masukkan yang berikut ini:

```
{
  "detail-type": [
    "customerCreated"
  ]
}
```

10. Pilih Selanjutnya.
11. Untuk jenis Target, pilih AWSlayanan.
12. Untuk Pilih target, pilih topik SNS dari daftar drop-down.
13. Untuk topik, pilih topik Amazon SNS yang Anda buat di langkah 1. Untuk tutorial ini, pilih EventBridge-IT-Test.
14. Untuk Pengaturan tambahan, lakukan hal berikut:
    a. Untuk Konfigurasikan input target, pilih Transformator input dari daftar drop-down.
    b. Pilih Konfigurasi transformator input
    c. untuk acara Contoh, masukkan berikut ini:

```
{
  "detail-type": "customerCreated"
}
```

    d. Untuk transformator masukan Target lakukan hal berikut:
        i. Untuk Input Path, masukkan berikut:

```
{"detail-type":"$.detail-type"}
```

ii. Untuk Template input, masukkan berikut:

```
"This event was of <detail-type> type."
```

e. Pilih Konfirmasi. .

15. Pilih Selanjutnya.
16. Pilih Selanjutnya.
17. Tinjau detail aturan dan pilih Buat aturan.

Langkah 4: Kirim peristiwa uji

Sekarang setelah Anda menyiapkan topik SNS dan aturannya, kami akan mengirimkan peristiwa pengujian untuk memastikan aturan berfungsi dengan benar.

Untuk mengirim peristiwa pengujian (konsol)

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Bus peristiwa.
3. Di ubin bus acara default, pilih Tindakan, Kirim acara.
4. Masukkan sumber peristiwa. Sebagai contoh, `TestEvent`.
5. Untuk jenis Detail, masukkan`customerCreated`.
6. Untuk detail Acara, masukkan`{}`.
7. Pilih Kirim.

Langkah 5: Mengonfirmasi keberhasilan

Jika Anda mendapatkan email dariAWS notifikasi yang cocok dengan output yang diharapkan, Anda telah berhasil menyelesaikan tutorial.

Langkah 6: Bersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankan itu. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus topik SNS

1. Buka halaman Topik konsol SNS.
2. Pilih topik yang Anda buat.
3. Pilih Delete (Hapus).
4. Masukkan **`delete me`**.
5. Pilih Delete (Hapus).

Untuk menghapus langganan SNS

1. Buka halaman Langganan konsol SNS.
2. Pilih langganan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Amazon EventBridge tutorial untuk mengintegrasikan dengan lainnyaAWSjasa

Amazon EventBridge bekerja denganAWSlayanan untuk diprosesacaraatau memanggilAWSsumber daya sebagaitargetdariaturan. Tutorial berikut menunjukkan kepada Anda cara mengintegrasikan EventBridge dengan lainnyaAWSservices.

Tutorial:

- Tutorial: Log status grup Auto Scaling menggunakan EventBridge
- Tutorial: Log panggilanAWS API menggunakan EventBridge
- Tutorial: Log status instans Amazon EC2 menggunakan EventBridge
- Tutorial: Operasi tingkat objek Log Amazon S3 EventBridge
- Tutorial: Mengirim kejadian ke aliran Amazon Kinesis menggunakan EventBridge danaws.events skema
- Tutorial: Jadwalkan snapshot Amazon EBS otomatis menggunakan EventBridge
- Tutorial: Kirim notifikasi saat objek Amazon S3 dibuat
- Tutorial: JadwalkanAWS Lambda fungsi menggunakan EventBridge

Tutorial: Log status grup Auto Scaling menggunakan EventBridge

Anda dapat menjalankan fungsi AWS Lambda yang mencatat peristiwa ketika grup Auto Scaling meluncurkan atau menghentikan instans Amazon EC2 yang menunjukkan apakah peristiwa berhasil.

Untuk informasi tentang skenario lainnya yang menggunakan peristiwa Amazon EC2 Auto Scaling, lihat Menggunakan EventBridge untuk menangani peristiwa Auto Scaling dalam Panduan Pengguna Amazon EC2 Auto Scaling.

Dalam tutorial ini, Anda membuat fungsi Lambda, dan Anda membuat aturan di EventBridge konsol yang memanggil fungsi ketika grup Amazon EC2 Auto Scaling meluncurkan atau menghentikan instans.

Langkah:

- Prasyarat
- Langkah 1: Membuat fungsi Lambda
- Langkah 2: Membuat aturan
- Langkah 3: Menguji aturan
- Langkah 4: Mengonfirmasi keberhasilan
- Langkah 5: Membersihkan sumber daya Anda

Prasyarat

Untuk menyelesaikan tutorial ini, Anda memerlukan sumber daya berikut:

- Grup Auto Scaling. Untuk informasi lebih lanjut tentang membuat grup Auto Scaling menggunakan konfigurasi peluncuran di Panduan Pengguna Amazon EC2 Auto Scaling.

Langkah 1: Membuat fungsi Lambda

Buat fungsi Lambda untuk log menskalakan keluar dan menskalaan kedalam peristiwa untuk grup Auto Scaling Anda.

Untuk membuat fungsi Lambda

1. Buka AWS Lambda konsol tersebut di https://console.aws.amazon.com/lambda/.
2. Pilih Buat fungsi.

3. Pilih Tulis dari awal.
4. Masukkan nama dan deskripsi untuk fungsi Lambda. Misalnya, beri nama fungsinya `LogAutoScalingEvent`.
5. Tinggalkan sisa pilihan sebagai default dan pilih Buat fungsi.
6. Pada tab Kode dari halaman fungsi, klik dua kali index.js.
7. Ganti kode yang ada dengan kode berikut.

```
'use strict';

exports.handler = (event, context, callback) => {
    console.log('LogAutoScalingEvent');
    console.log('Received event:', JSON.stringify(event, null, 2));
    callback(null, 'Finished');
};
```

8. Pilih Deploy.

Langkah 2: Membuat aturan

Buat aturan untuk menjalankan fungsi Lambda yang Anda buat di Langkah 1. Aturan berjalan saat grup Auto Scaling Anda memulai atau menghentikan sebuah instans.

Untuk membuat tabel

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan. Misalnya, beri nama aturan`TestRule`
5. Untuk bus Events, pilih bus kejadian yang ingin Anda kaitkan dengan aturan ini. Jika Anda ingin aturan ini cocok dengan kejadian yang berasal dari akun Anda, pilih default. Saat layanan AWS di akun Anda menghasilkan kejadian, layanan tersebut akan selalu masuk ke bus kejadian default akun Anda.
6. Untuk jenis Aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.
8. Untuk sumber acara, pilih AWSlayanan.
9. Untuk pola Event, lakukan hal berikut:

   a. Untuk Sumber peristiwa, pilih Auto Scaling dari daftar drop-down.
   b. Untuk Jenis acara, pilih Instance Launch dan Terminate dari daftar drop-down.
   c. Pilih Setiap kejadian instance dan Setiap nama grup.
10. Pilih Selanjutnya.
11. Untuk jenis Target, pilih AWSlayanan.
12. Untuk Pilih target, pilih fungsi Lambda dari daftar drop-down.
13. Untuk Fungsi, pilih fungsi Lambda yang Anda buat di bagian fungsi Lambda: Buat fungsi Lambda. Dalam contoh ini, pilih`LogAutoScalingEvent`.
14. Pilih Selanjutnya.
15. Pilih Selanjutnya.
16. Tinjau detail aturan dan pilih Buat aturan.

Langkah 3: Menguji aturan

Anda dapat menguji aturan Anda dengan penskalaan grup Auto Scaling secara manual sehingga meluncurkan sebuah instans. Tunggu beberapa menit hingga peristiwa penskalaan keluar terjadi, lalu verifikasi bahwa fungsi Lambda Anda telah dipanggil.

Untuk menguji aturan menggunakan grup Auto Scaling

1. Untuk menambah ukuran grup Auto Scaling Anda, lakukan hal berikut:
   a. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
   b. Dalam panel navigasi, di bawah Auto Scaling, Grup Auto Scaling.
   c. Pilih kotak centang di samping grup Auto Scaling Anda.
   d. Pada tab Detail, pilih Edit. Untuk Diinginkan, tingkatkan kapasitas yang diinginkan sebesar satu. Misalnya, jika nilai saat ini adalah 2, masukkan 3. Kapasitas yang diinginkan harus kurang dari atau sama dengan ukuran maksimum grup. Jika nilai baru Anda untuk Diinginkan lebih besar dari Maks, Anda harus memperbarui Maks. Setelah selesai, pilih Simpan.
2. Untuk melihat output dari fungsi Lambda Anda, lakukan hal berikut:
   a. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
   b. Di panel navigasi, pilih Log.

c. Pilih nama grup catatan untuk fungsi Lambda Anda (`/aws/lambda/function-name`).

d. Pilih nama aliran log untuk melihat data yang disediakan oleh fungsi untuk instans yang Anda jalankan.

3. (Opsional) Setelah selesai, Anda dapat mengurangi kapasitas yang diinginkan per satu sehingga grup Auto Scaling kembali ke ukuran sebelumnya.

Langkah 4: Mengonfirmasi keberhasilan

Jika Anda melihat peristiwa Lambda di CloudWatch Logs, Anda telah berhasil menyelesaikan tutorial ini. Jika peristiwa tersebut tidak ada di CloudWatch Logs (Logs), mulai pemecahan masalah dengan memverifikasi aturan berhasil dibuat dan, jika aturan terlihat benar, verifikasi kode fungsi Lambda Anda benar.

Langkah 5: Membersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankannya. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Untuk menghapus fungsi Lambda

1. Buka halaman Fungsi di konsol Lambda.
2. Pilih fungsi yang Anda buat.
3. Pilih Tindakan, Hapus.
4. Pilih Hapus.

Tutorial: Log panggilanAWS API menggunakan EventBridge

Anda dapat menggunakan EventBridge aturan Amazon untuk bereaksi terhadap panggilan API yang dibuat olehAWS layanan yang direkam olehAWS CloudTrail.

Dalam tutorial ini, Anda membuat AWS CloudTrailjejak, fungsi Lambda, dan aturan di EventBridge konsol. Aturan memanggil fungsi Lambda ketika instans Amazon EC2 dihentikan.

Langkah:

- Langkah 1: Buat AWS CloudTrail jejak
- Langkah 2: Buat AWS Lambda fungsi
- Langkah 3: Buat aturan
- Langkah 4: Uji aturan
- Langkah 5: Mengonfirmasi keberhasilan
- Langkah 6: Bersihkan sumber daya Anda

Langkah 1: Buat AWS CloudTrail jejak

Jika Anda sudah memiliki jejak diatur, lewati ke langkah 2.

Untuk membuat jejak

1. Buka CloudTrail konsol di https://console.aws.amazon.com/cloudtrail/.
2. Pilih Jejak, Buat jejak.
3. Untuk Nama jejak, ketikkan nama untuk jejak.
4. Untuk Lokasi penyimpanan, di Buat bucket S3 baru.
5. Untuk AWS KMS alias, ketik alias untuk bukti kunci KMS.
6. Pilih Selanjutnya.
7. Pilih Selanjutnya.
8. Pilih Buat jejak.

Langkah 2: Buat AWS Lambda fungsi

Buat fungsi Lambda untuk log peristiwa panggilan API.

Untuk membuat fungsi Lambda

1. Buka AWS Lambda konsol tersebut di https://console.aws.amazon.com/lambda/.
2. Pilih Buat fungsi.
3. Pilih Tulis dari scratch.
4. Masukkan nama dan deskripsi untuk fungsi Lambda. Misalnya, beri nama fungsi tersebut `LogEC2StopInstance`.
5. Tinggalkan sisa pilihan sebagai default dan pilih Buat fungsi.
6. Pada tab Kode dari halaman fungsi, klik dua kali index.js.
7. Ganti kode yang ada dengan kode berikut.

```
'use strict';

exports.handler = (event, context, callback) => {
    console.log('LogEC2StopInstance');
    console.log('Received event:', JSON.stringify(event, null, 2));
    callback(null, 'Finished');
};
```

8. Pilih Men-deploy.

Langkah 3: Buat aturan

Buat aturan untuk menjalankan fungsi Lambda Anda buat di langkah 2 setiap kali Anda berhenti instans Amazon EC2.

Untuk membuat tabel

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan. Misalnya, beri nama aturan`TestRule`
5. Untuk bus kejadian, pilih bus kejadian yang ingin Anda kaitkan dengan aturan ini. Jika Anda ingin aturan ini cocok dengan peristiwa yang berasal dari akun Anda, pilih default. Saat layanan AWS di akun Anda menghasilkan kejadian, layanan tersebut akan selalu masuk ke bus kejadian default akun Anda.
6. Untuk jenis Aturan, pilih Aturan dengan pola peristiwa.

7. Pilih Selanjutnya.
8. Untuk sumber Event, pilih AWSlayanan.
9. Untuk Pola peristiwa, lakukan hal berikut:
    a. Untuk Sumber acara, pilih EC2 dari daftar drop-down.
    b. Untuk jenis Event, pilih AWSAPI Call via CloudTrail dari daftar drop-down.
    c. Pilih Operasi spesifik dan masukkan`StopInstances`.
10. Pilih Selanjutnya.
11. Untuk jenis Target, pilih AWSlayanan.
12. Untuk Pilih target, pilih fungsi Lambda dari daftar drop-down.
13. Untuk Fungsi, pilih fungsi Lambda yang Anda buat. di bagian fungsi Lambda Langkah 1: Buat fungsi Lambda. Dalam contoh ini, pilih`LogEC2StopInstance`.
14. Pilih Selanjutnya.
15. Pilih Selanjutnya.
16. Tinjau detail aturan dan pilih Buat aturan.

Langkah 4: Uji aturan

Anda dapat menguji aturan Anda dengan pemfilteran stopword instans Amazon EC2 menggunakan konsol Amazon EC2. Tunggu beberapa menit hingga instans berhenti, lalu periksaAWS Lambda metrik Anda di CloudWatch konsol untuk memverifikasi bahwa fungsi Anda berjalan.

Untuk menguji aturan Anda dengan pemfilteran stopword instans

1. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
2. Peluncuran instans. Untuk informasi selengkapnya, lihat Peluncuran Instans Anda di Panduan Pengguna Amazon EC2 untuk Instans Linux.
3. Hentikan instans. Untuk informasi lebih lanjut, lihat Hentikan dan Mulai Instans Anda dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.
4. Untuk melihat output dari fungsi Lambda Anda, lakukan hal berikut:
    a. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
    b. Di panel navigasi, pilih Log.
    c. Pilih nama grup log untuk fungsi Lambda Anda (`/aws/lambda/`*`function-name`*).

d. Pilih nama pengaliran log untuk melihat data yang disediakan oleh fungsi untuk instans yang Anda hentikan.

5. (Opsional) Ketika Anda selesai, akhiri instans yang dihentikan. Untuk informasi lebih lanjut, lihat Akhiri Instans Anda dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

Langkah 5: Mengonfirmasi keberhasilan

Jika Anda melihat peristiwa Lambda di CloudWatch Logs, Anda telah berhasil menyelesaikan tutorial ini. Jika peristiwa tersebut tidak ada di CloudWatch Logs (Clogs), mulai pemecahan masalah dengan memverifikasi aturan berhasil dibuat dan, jika aturan terlihat benar, verifikasi kode fungsi Lambda Anda benar.

Langkah 6: Bersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankan itu. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Untuk menghapus fungsi Lambda

1. Buka halaman Fungsi di konsol Lambda.
2. Pilih fungsi yang Anda buat.
3. Pilih Tindakan, Hapus.
4. Pilih Hapus.

Untuk menghapus CloudTrail jejak

1. Buka halaman Trails CloudTrail konsol.
2. Pilih jejak yang Anda buat.

3. Pilih Hapus.
4. Pilih Hapus.

Tutorial: Log status instans Amazon EC2 menggunakan EventBridge

Anda dapat membuat fungsi AWS Lambda yang mencatat perubahan status untuk instans Amazon EC2. Setelah itu, Anda dapat membuat aturan yang menjalankan fungsi Lambda Anda setiap kali ada transisi status atau transisi ke satu atau beberapa status yang menarik. Dalam tutorial ini, Anda log peluncuran setiap instans baru.

Langkah-langkah:

- Langkah 1: Membuat Fungsi AWS Lambda
- Langkah 2: Membuat aturan
- Langkah 3: Menguji aturan
- Langkah 4: Mengonfirmasi keberhasilan
- Langkah 5: Membersihkan sumber daya Anda

Langkah 1: Membuat Fungsi AWS Lambda

Buat fungsi Lambda untuk log peristiwa perubahan status Saat Anda membuat aturan di Langkah 2, Anda menentukan fungsi ini.

Untuk membuat fungsi Lambda

1. Buka AWS Lambda konsol tersebut di https://console.aws.amazon.com/lambda/.
2. Pilih Buat fungsi.
3. Pilih Tulis dari scratch.
4. Masukkan nama dan deskripsi untuk fungsi Lambda. Misalnya, beri nama fungsi tersebut `LogEC2InstanceStateChange`.
5. Tinggalkan sisa pilihan sebagai default dan pilih Buat fungsi.
6. Pada tab Kode dari halaman fungsi, klik dua kali index.js.
7. Ganti kode yang ada dengan kode berikut.

```
'use strict';

exports.handler = (event, context, callback) => {
    console.log('LogEC2InstanceStateChange');
    console.log('Received event:', JSON.stringify(event, null, 2));
    callback(null, 'Finished');
```

```
};
```

8. Pilih Deploy.

Langkah 2: Membuat aturan

Membuat aturan untuk menjalankan fungsi Lambda yang Anda buat di Langkah 1. Aturan berjalan ketika Anda meluncurkan instans Amazon EC2.

Untuk membuat EventBridge aturan

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan. Misalnya, beri nama aturan`TestRule`
5. Untuk bus kejadian, pilih bus kejadian yang ingin Anda kaitkan dengan aturan ini. Jika Anda ingin aturan ini cocok dengan kejadian yang berasal dari akun Anda, pilih default. Saat layanan AWS di akun Anda menghasilkan kejadian, layanan tersebut akan selalu masuk ke bus kejadian default akun Anda.
6. Untuk jenis Aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.
8. Untuk sumber Event, pilih AWSlayanan.
9. Untuk pola peristiwa, lakukan hal berikut:
    a. Untuk Sumber acara, pilih EC2 dari daftar drop-down.
    b. Untuk Jenis acara, pilih Pemberitahuan Perubahan Status Instans EC2 dari daftar drop-down.
    c. Pilih Status spesifik dan pilih menjalankan dari daftar drop-down.
    d. Pilih Instans apa pun
10. Pilih Selanjutnya.
11. Untuk jenis Target, pilih AWSlayanan.
12. Untuk Pilih target, pilih fungsi Lambda dari daftar drop-down.
13. Untuk Fungsi, pilih fungsi Lambda yang Anda buat di bagian fungsi Lambda yang Anda buat di Langkah 1: Buat fungsi Lambda. Dalam contoh ini, pilih`LogEC2InstanceStateChange`.
14. Pilih Selanjutnya.

15. Pilih Selanjutnya.
16. Tinjau detail aturan dan pilih Buat aturan.

Langkah 3: Menguji aturan

Anda dapat menguji aturan Anda dengan pemfilteran stopword instans Amazon EC2 menggunakan konsol Amazon EC2. Tunggu beberapa menit hingga instans berhenti, lalu periksaAWS Lambda metrik Anda di CloudWatch konsol untuk memverifikasi bahwa fungsi Anda berjalan.

Untuk menguji aturan Anda dengan pemfilteran stopword instans

1. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
2. Peluncuran instans. Untuk informasi selengkapnya, lihat Peluncuran Instans Anda di Panduan Pengguna Amazon EC2 untuk Instans Linux.
3. Hentikan instans. Untuk informasi lebih lanjut, lihat Hentikan dan Mulai Instans Anda dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.
4. Untuk melihat output dari fungsi Lambda Anda, lakukan hal berikut:
    a. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
    b. Di panel navigasi, pilih Log.
    c. Pilih nama grup log untuk fungsi Lambda Anda (`/aws/lambda/`*`function-name`*).
    d. Pilih nama pengaliran log untuk melihat data yang disediakan oleh fungsi untuk instans yang Anda hentikan.
5. (Opsional) Ketika Anda selesai, akhiri instans yang dihentikan. Untuk informasi lebih lanjut, lihat Akhiri Instans Anda dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

Langkah 4: Mengonfirmasi keberhasilan

Jika Anda melihat peristiwa Lambda di CloudWatch Logs, Anda telah berhasil menyelesaikan tutorial ini. Jika peristiwa tersebut tidak ada di CloudWatch Logs (Cs), mulai pemecahan masalah dengan memverifikasi aturan berhasil dibuat dan, jika aturan terlihat benar, verifikasi kode fungsi Lambda Anda benar.

Langkah 5: Membersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankannya. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Untuk menghapus fungsi Lambda

1. Buka halaman Fungsi di konsol Lambda.
2. Pilih fungsi yang Anda buat.
3. Pilih Tindakan, Hapus.
4. Pilih Hapus.

Tutorial: Operasi tingkat objek Log Amazon S3 EventBridge

Anda dapat mencatat operasi API tingkat objek pada bucket Amazon S3 Sebelum Amazon EventBridge dapat mencocokkan kejadian, Anda harus menggunakan AWS CloudTrailuntuk mengatur dan mengkonfigurasi jejak untuk menerima kejadian ini.

Dalam tutorial ini, Anda CloudTrail membuat AWS Lambdafungsi, dan kemudian membuat fungsi di EventBridge konsol yang memanggil fungsi tersebut untuk merespons kejadian data S3.

Langkah:

- Langkah 1: Konfigurasi jejak AWS CloudTrail
- Langkah 2: Buat fungsi AWS Lambda
- Langkah 3: Buat Aturan
- Langkah 4: Uji Aturan
- Langkah 5: Mengonfirmasi keberhasilan
- Langkah 6: Bersihkan sumber daya Anda

Langkah 1: Konfigurasi jejak AWS CloudTrail

Untuk mencatat kejadian data untuk bucket S3 keAWS CloudTrail dan EventBridge, pertama Anda membuat jejak. Jejak menangkap panggilan API dan kejadian terkait di akun Anda dan kemudian mengirimkan file log ke bucket S3 yang Anda tentukan. Anda dapat memperbarui jejak yang ada atau membuat jejak.

Untuk informasi selengkapnya, lihat Kejadian Data dalam AWS CloudTrailPanduan Pengguna.

Untuk membuat jejak

1. Buka CloudTrail konsol di https://console.aws.amazon.com/cloudtrail/.
2. Pilih Jejak, Buat jejak.
3. Untuk Nama jejak, ketikkan nama untuk jejak.
4. Untuk Lokasi penyimpanan, di Buat bucket S3 baru.
5. Untuk AWS KMS alias, ketik alias untuk bukti kunci KMS.
6. Pilih Berikutnya.
7. Untuk Jenis kejadian, pilihKejadian data

8. Untuk Kejadian data,, lakukan salah satu langkah berikut:

   - Untuk mencatat kejadian data untuk semua objek Amazon S3 dalam bucket, tentukan bucket S3 dan prefiks kosong. Ketika suatu kejadian terjadi pada sebuah objek di bucket tersebut, jejak meproses dan mencatat kejadian.
   - Untuk mencatat kejadian data untuk objek tertentu Amazon S3 dalam bucket, tentukan bucket S3 dan prefiks objek. Ketika suatu kejadian terjadi pada sebuah objek di bucket tersebut dan objek dimulai dengan prefiks yang ditentukan, jejak memproses dan mencatat kejadian tersebut.
9. Untuk setiap sumber daya, pilih apakah Anda akan mencatat kejadian Baca, kejadian Tulis, atau keduanya.
10. Pilih Berikutnya.
11. Pilih Buat jejak.

Langkah 2: Buat fungsi AWS Lambda

Buat fungsi Lambda untuk mencatat kejadian data untuk bucket S3 Anda.

Untuk membuat fungsi Lambda

1. Buka AWS Lambda konsol tersebut di https://console.aws.amazon.com/lambda/.
2. Pilih Buat fungsi.
3. Pilih Tulis dari scratch.
4. Masukkan nama dan deskripsi untuk fungsi Lambda. Misalnya, beri nama fungsi tersebut `LogS3DataEvents`.
5. Tinggalkan sisa pilihan sebagai default dan pilih Buat fungsi.
6. Pada tab Kode dari halaman fungsi, klik dua kali index.js.
7. Ganti kode yang ada dengan kode berikut.

```
'use strict';

exports.handler = (event, context, callback) => {
    console.log('LogS3DataEvents');
    console.log('Received event:', JSON.stringify(event, null, 2));
    callback(null, 'Finished');
};
```

8. Pilih Men-deploy.

Langkah 3: Buat Aturan

Buat aturan untuk menjalankan fungsi Lambda yang Anda buat di Langkah 2. Aturan ini berjalan sebagai respons kejadian data Amazon S3.

Untuk membuat tabel

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan. Misalnya, beri nama aturan`TestRule`
5. Untuk bus Events, pilih bus kejadian yang ingin Anda kaitkan dengan aturan ini. Jika Anda ingin aturan ini cocok dengan peristiwa yang berasal dari akun Anda, pilih default. Saat layanan AWS di akun Anda menghasilkan kejadian, layanan tersebut akan selalu masuk ke bus kejadian default akun Anda.
6. Untuk jenis Aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.
8. Untuk sumber acara, pilih AWSlayanan.
9. Untuk pola Event, lakukan hal berikut:
    a. Untuk Sumber acara, pilih Simple Storage Service (S3) dari daftar drop-down.
    b. Untuk jenis Event, pilih Object-Level API call via CloudTrail dari daftar drop-down.
    c. Pilih Operasi khusus, lalu pilih PutObject.
    d. Secara default, aturan sesuai dengan kejadian data untuk semua bucket di Region. Untuk mencocokkan kejadian data untuk bucket tertentu, pilihTentukan bucket berdasarkan nama dan masukkan satu atau beberapa bucket.
10. Pilih Selanjutnya.
11. Untuk jenis Target, pilih AWSlayanan.
12. Untuk Pilih target, pilih fungsi Lambda dari daftar drop-down.
13. Untuk Fungsi, pilih fungsi`LogS3DataEvents` Lambda yang Anda di langkah 1.
14. Pilih Selanjutnya.
15. Pilih Selanjutnya.

16. Tinjau detail aturan dan pilih Buat aturan.

Langkah 4: Uji Aturan

Untuk menguji aturan, letakkan objek di bucket S3 Anda. Anda dapat memverifikasi bahwa fungsi Lambda Anda dipanggil.

Untuk menampilkan catatan fungsi Lambda Anda

1. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Di panel navigasi, pilih Log.
3. Pilih nama grup catatan untuk fungsi Lambda Anda (`/aws/lambda/`*`function-name`*).
4. Pilih nama aliran catatan untuk menampilkan data yang disediakan oleh fungsi untuk instans yang Anda luncurkan.

Anda juga dapat memeriksa CloudTrail catatan Anda di bucket S3 yang Anda tentukan untuk jejak Anda. Untuk informasi selengkapnya, lihat Mendapatkan dan Melihat CloudTrail Log Files Anda di PanduanAWS CloudTrail Pengguna.

Langkah 5: Mengonfirmasi keberhasilan

Jika Anda melihat peristiwa Lambda di CloudWatch log, Anda telah berhasil menyelesaikan tutorial ini. Jika peristiwa tersebut tidak ada di CloudWatch Log (Lambda), mulai pemecahan masalah aturan berhasil dibuat dan, jika aturan terlihat benar, verifikasi kode fungsi Lambda Anda benar.

Langkah 6: Bersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankan itu. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Untuk menghapus fungsi Lambda

1. Buka halaman Fungsi di konsol Lambda.
2. Pilih fungsi yang Anda buat.
3. Pilih Tindakan, Hapus.
4. Pilih Hapus.

Untuk menghapus CloudTrail jejak

1. Buka halaman Trails CloudTrail konsol.
2. Pilih jenis yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Tutorial: Mengirim kejadian ke aliran Amazon Kinesis menggunakan EventBridge dan`aws.events` skema

Anda dapatAWS mengirim kejadian EventBridge ke aliran Amazon Kinesis, membuat aplikasi Kinesis Data Streams, dan memproses data dalam jumlah besar. Dalam tutorial ini, Anda membuat aliran Kinesis, lalu membuat aturan di EventBridge konsol yang mengirimkan kejadian ke aliran tersebut ketika instans Amazon EC2 berhenti.

Langkah:

- Prasyarat
- Langkah 1: Buat aliran Amazon Kinesis
- Langkah 2: Buat aturan
- Langkah 3: Uji aturan
- Langkah 4: Verifikasi bahwa kejadian telah dikirim
- Langkah 5: Bersihkan sumber daya Anda

Prasyarat

Dalam tutorial ini, Anda akan menggunakan berikut ini:

- GunakanAWS CLI untuk bekerja dengan aliran Kinesis.

  Untuk menginstal AWS CLI, lihat Menginstal, memperbarui, dan menghapus instalasi AWS CLI versi 2.

> **Note**
>
> Tutorial ini menggunakanAWS peristiwa dan dibangun di`aws.events` skema registri. Anda juga dapat membuat EventBridge aturan berdasarkan skema peristiwa kustom Anda dengan menambahkannya ke registri skema kustom secara manual, atau dengan menggunakan penemuan skema.
> Untuk informasi selengkapnya tentang skema, lihat ???. Untuk informasi selengkapnya tentang membuat aturan menggunakan opsi pola acara lainnya, lihat???.

Langkah 1: Buat aliran Amazon Kinesis

Untuk membuat aliran, pada prompt perintah, gunakan perintah `create-stream` AWS CLI.

```
aws kinesis create-stream --stream-name test --shard-count 1
```

Ketika status aliran adalah `ACTIVE`, aliran sudah siap. Untuk memeriksa status aliran, gunakan perintah `describe-stream`.

```
aws kinesis describe-stream --stream-name test
```

Langkah 2: Buat aturan

Buat aturan untuk mengirim kejadian ke aliran Anda ketika Anda menghentikan instans Amazon EC2.

Untuk membuat aturan

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan. Misalnya, beri nama aturannya`TestRule`
5. Untuk Event bus, pilih default.
6. Untuk jenis Aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.
8. Untuk Sumber acara, pilih AWSacara atau acara EventBridge mitra.
9. Untuk Metode pembuatan, pilih Gunakan skema.
10. Untuk pola kejadian, lakukan hal berikut:

    a. Untuk Jenis skema, pilih Pilih skema dari registri Skema.

    b. Untuk registri Skema, pilih aws.events dari daftar drop-down.

    c. Untuk Skema, pilih aws.ec2 @EC2InstanceStateChangeNotification dari daftar drop-down.

    EventBridge menampilkan skema acara di bawah Model.

    EventBridge menampilkan tanda bintang merah di samping properti apa pun yang diperlukan untuk acara tersebut, bukan untuk pola acara.

    d. Dalam Model, mengatur properti filter acara berikut:

i. Pilih+Edit di samping properti negara.

Tinggalkan Hubungan kosong. Untuk Nilai, masukkan `running`. Pilih Set.

ii. Pilih+Edit di samping properti sumber.

Tinggalkan Hubungan kosong. Untuk Nilai, masukkan `aws.ec2`. Pilih Set.

iii. Pilih+Edit di samping properti detail-type.

Tinggalkan Hubungan kosong. Untuk Nilai, masukkan `EC2 Instance State-change Notification`. Pilih Set.

e. Untuk melihat pola peristiwa yang telah Anda buat, pilih Generate event pattern di JSON

EventBridge menampilkan pola acara di JSON:

```
{
  "detail": {
    "state": ["running"]
  },
  "detail-type": ["EC2 Instance State-change Notification"],
  "source": ["aws.ec2"]
}
```

11. Pilih Selanjutnya.
12. Untuk jenis Target, pilih AWSlayanan.
13. Untuk Pilih target, pilih aliran Kinesis dari daftar drop-down.
14. Untuk Stream, pilih aliran Kinesis yang Anda buat di Langkah 1: Buat bagian aliran Amazon Kinesis. Dalam contoh ini, pilih`test`.
15. Untuk peran Eksekusi, pilih Buat peran baru untuk sumber daya khusus ini.
16. Pilih Selanjutnya.
17. Pilih Selanjutnya.
18. Tinjau detail aturan dan pilih Buat aturan.

Langkah 3: Uji aturan

Untuk menguji aturan Anda, hentikan satu instans Amazon EC2. Tunggu beberapa menit hingga instans berhenti, lalu periksa CloudWatch metrik Anda untuk memverifikasi bahwa fungsi Anda berjalan.

Untuk menguji aturan Anda dengan pemfilteran stopword instans

1. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
2. Peluncuran instans. Untuk informasi selengkapnya, lihat Peluncuran Instans Anda di Panduan Pengguna Amazon EC2 untuk Instans Linux.
3. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
4. Di panel navigasi, pilih Aturan.

   Pilih nama aturan yang Anda buat dan pilihMetrik untuk aturan.
5. (Opsional) Setelah selesai, akhiri instans tersebut. Untuk informasi lebih lanjut, lihat Mengakhiri Instans Anda di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Langkah 4: Verifikasi bahwa kejadian telah dikirim

Anda dapat menggunakan AWS CLI untuk mendapatkan catatan dari aliran untuk memverifikasi bahwa kejadian telah dikirim.

Untuk mendapatkan catatan

1. Untuk mulai membaca dari aliran Kinesis Anda, pada prompt perintah, gunakan perintah `get-shard-iterator`.

   ```
   aws kinesis get-shard-iterator --shard-id shardId-000000000000 --shard-iterator-type TRIM_HORIZON --stream-name test
   ```

   Berikut ini adalah contoh output.

   ```
   {
       "ShardIterator": "AAAAAAAAAAHSywljv0zEgPX4NyKdZ5wryMzP9yALs8NeKbUjp1IxtZs1Sp+KEd9I6AJ9ZG4lNR1EMi+9Md/nHvtLyxpfhEzYvkTZ4D9DQVz/mBYWRO6OTZRKnW9gd+efGN2aHFdkH1rJl4BL9Wyrk+ghYG22D2T1Da2EyNSH1+LAbK33gQweTJADBdyMwlo5r6PqcP2dzhg="
   }
   ```
2. Untuk mendapatkan catatan, gunakan berikut perintah `get-records`. Gunakan serpihan iterator dari output pada langkah sebelumnya.

   ```
   aws kinesis get-records --shard-iterator AAAAAAAAAAHSywljv0zEgPX4NyKdZ5wryMzP9yALs8NeKbUjp1IxtZs1Sp
   ```

```
+KEd9I6AJ9ZG4lNR1EMi+9Md/nHvtLyxpfhEzYvkTZ4D9DQVz/mBYWRO6OTZRKnW9gd
+efGN2aHFdkH1rJl4BL9Wyrk+ghYG22D2T1Da2EyNSH1+LAbK33gQweTJADBdyMwlo5r6PqcP2dzhg=
```

Jika perintah berhasil, perintah akan meminta catatan dari aliran Anda untuk shard yang ditentukan. Anda dapat menerim nol atau beberapa catatan. Catatan apa pun yang dikembalikan mungkin tidak mewakili semua catatan di aliran Anda. Jika Anda tidak menerima data yang Anda harapkan, pangill `get-records` secara terus menerus.

3. Catatan dalam Kinesis dikodekan dalam Base64. Gunakan decoder Base64 untuk memecahkan kode data sehingga Anda dapat memverifikasi bahwa kejadian tersebut adalah kejadian yang dikirim ke aliran dalam bentuk JSON.

Langkah 5: Bersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankannya. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Untuk menghapus aliran Kinesis

1. Buka Halaman data streams dari konsol Kinesis.
2. Pilih aliran yang Anda buat.
3. Pilih Tindakan, Hapus.
4. Masukkan menghapus di fiekd dan pilih Hapus.

Tutorial: Jadwalkan snapshot Amazon EBS otomatis menggunakan EventBridge

Anda dapat menjalankan EventBridge aturan sesuai jadwal. Dalam tutorial ini, Anda membuat snapshot volume dari Amazon Elastic Block Store (Amazon EBS) yang ada sesuai jadwal. Anda dapat memilih nilai tetap untuk membuat snapshot setiap beberapa menit atau menggunakan ekspresi cron untuk membuat snapshot pada waktu tertentu dalam sehari.

> **Important**
>
> Untuk membuat peraturan dengan target bawaan, Anda harus menggunakanAWS Management Console.

Langkah-langkahnya:

- Langkah 1: Buat aturan
- Langkah 2: Uji aturannya
- Langkah 3: Konfirmasikan keberhasilan
- Langkah 4: Bersihkan sumber daya Anda

Langkah 1: Buat aturan

Buat aturan yang mengambil snapshot pada jadwal. Anda dapat menggunakan ekspresi nilai atau ekspresi cron untuk menentukan jadwal. Selengkapnya, lihat Membuat EventBridge aturan Amazon yang berjalan sesuai jadwal.

Untuk membuat aturan

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan.

   Aturan tidak boleh memiliki nama yang sama dengan aturan lain di Wilayah yang sama dan di bus kejadian yang sama.

5. Untuk Pilihan bus kejadian yang ingin Anda kaitkan dengan aturan ini. Jika Anda ingin aturan ini cocok dengan kejadian yang berasal dari akun Anda, pilih bus kejadian default AWS. Saat layanan AWS di akun Anda menghasilkan kejadian, layanan tersebut akan selalu masuk ke bus kejadian default akun Anda.
6. Untuk jenis Aturan, pilih Jadwal.
7. Pilih Selanjutnya.
8. Untuk pola Jadwal, pilih Jadwal yang berjalan dengan tarif reguler, seperti setiap 10 menit. dan masuk**5** dan pilih Menit dari daftar tarik-turun.
9. Pilih Selanjutnya.
10. Untuk jenis Target, pilih AWSlayanan.
11. Untuk Pilih target, pilih EBS Create Snapshot dari daftar drop-down.
12. Untuk ID Volume, masukkan ID volume dari volume Amazon EBS.
13. Untuk peran Eksekusi, pilih Buat peran baru untuk sumber daya khusus ini.
14. Pilih Selanjutnya.
15. Pilih Selanjutnya.
16. Tinjau detail aturan dan pilih Buat aturan.

Langkah 2: Uji aturannya

Anda dapat memverifikasi bahwa aturan berfungsi dengan melihat snapshot pertama Anda setelah diambil.

Untuk menguji aturan Anda

1. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
2. Di panel navigasi, di bagian Elastic Block Store, pilih Snapshot.
3. Verifikasi bahwa snapshot pertama muncul di daftar.

Langkah 3: Konfirmasikan keberhasilan

Jika melihat snapshot dalam daftar, Anda telah berhasil menyelesaikan tutorial ini. Jika snapshot tidak ada dalam daftar, mulai pemecahan masalah dengan memverifikasi aturan berhasil dibuat.

Langkah 4: Bersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankannya. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Tutorial: Kirim notifikasi saat objek Amazon S3 dibuat

Anda dapat mengirim notifikasi email ketika objek Amazon Simple Storage Service (Amazon S3) dibuat menggunakan Amazon EventBridge dan Amazon SNS. Dalam tutorial ini, Anda akan membuat topik SNS dan berlangganan. Kemudian, Anda akan membuat aturan di EventBridge konsol yang mengirimkan peristiwa ke topik tersebut saat`Object Created` acara Amazon S3 diterima.

Langkah:

- Prasyarat
- Langkah 1: Membuat topik Amazon SNS
- Langkah 2: Membuat langganan Amazon SNS
- Langkah 3: Buat aturan
- Langkah 4: Uji aturan
- Langkah 5: Membersihkan sumber daya Anda

Prasyarat

Untuk menerima acara Amazon S3 EventBridge, Anda harus mengaktifkan EventBridge di konsol Amazon S3. Tutorial ini mengasumsikan EventBridge diaktifkan. Untuk informasi selengkapnya, lihat Mengaktifkan Amazon EventBridge di konsol S3.

Langkah 1: Membuat topik Amazon SNS

Buat topik untuk menerima acara dari EventBridge.

Cara membuat topik

1. Buka konsol Amazon SNS di https://console.aws.amazon.com/sns/v3/home.
2. Di panel navigasi, pilih Topik.
3. Pilih Buat topik.
4. Untuk Tipe, pilih Standar.
5. Masukkan**`eventbridge-test`** sebagai nama topik.
6. Pilih Buat topik.

Langkah 2: Membuat langganan Amazon SNS

Buat langganan untuk mendapatkan notifikasi email dari Amazon S3 saat acara diterima oleh topik.

Untuk membuat langganan

1. Buka konsol Amazon SNS di https://console.aws.amazon.com/sns/v3/home.
2. Di panel navigasi, pilih Subscriptions (Langganan).
3. Pilih Create subscription (Buat langganan).
4. Untuk topik ARN, pilih topik yang Anda buat di langkah 1. Untuk tutorial ini, pilih eventbridge-test.
5. Untuk Protokol, pilih Email.
6. Untuk Titik Akhir, masukkan alamat email Anda.
7. Pilih Buat langganan.
8. Konfirmasikan langganan dengan memilih Konfirmasi langganan di email yang Anda terima dariAWS notifikasi.

Langkah 3: Buat aturan

Membuat aturan untuk mengirim peristiwa ke topik Anda ketika objek Amazon S3 dibuat.

Untuk membuat tabel

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan. Misalnya, beri nama aturan`s3-test`
5. Untuk Event bus, pilih default.
6. Untuk jenis Aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.
8. Untuk Sumber acara, pilih AWSacara atau acara EventBridge mitra.
9. Untuk Metode pembuatan, pilih Gunakan bentuk pola.
10. Untuk pola peristiwa, lakukan hal berikut:
    a. Untuk Sumber peristiwa, pilih AWSlayanan dari daftar drop-down.
    b. Untuk AWSlayanan, pilih Simple Storage Service (S3) dari daftar drop-down.

c. Untuk jenis Event, pilih Amazon S3 Event Notification dari daftar drop-down.
d. Pilih Specific events (s) dan pilih Object Created dari daftar drop-down.
e. Pilih ember apa saja

11. Pilih Selanjutnya.
12. Untuk jenis Target, pilih AWSlayanan.
13. Untuk Pilih target, pilih topik SNS dari daftar drop-down.
14. Untuk Topik, pilih topik Amazon SNS yang Anda buat di Langkah 1: Buat bagian topik SNS. Dalam contoh ini, pilih`eventbridge-test`.
15. Pilih Selanjutnya.
16. Pilih Selanjutnya.
17. Tinjau detail aturan dan pilih Buat aturan.

Langkah 4: Uji aturan

Untuk menguji aturan Anda, buat objek Amazon S3 dengan mengunggah file ke bucket EventBridge yang diaktifkan. Kemudian, tunggu beberapa menit dan verifikasi apakah Anda menerima email dariAWS notifikasi.

Langkah 5: Membersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankannya. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus topik SNS

1. Buka halaman Topik konsol SNS.
2. Pilih topik yang Anda buat.
3. Pilih Delete (Hapus).
4. Masukkan **`delete me`**.
5. Pilih Delete (Hapus).

Untuk menghapus langganan SNS

1. Buka halaman Langganan konsol SNS.

2. Pilih langganan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Tutorial: JadwalkanAWS Lambda fungsi menggunakan EventBridge

Anda dapat mengatur aturan untuk menjalankan AWS Lambda fungsi pada jadwal. Tutorial ini menunjukkan cara menggunakan AWS Management Console atau AWS CLI atau membuat aturan. Jika Anda ingin menggunakan AWS CLI tapi belum menginstalnya, lihat Menginstal, memperbarui, dan menghapus instalasiAWS CLIversi 2.

Untuk jadwal, EventBridge tidak memberikan presisi tingkat kedua di ekspresi jadwal. Resolusi terbaik yang menggunakan ekspresi cron adalah satu menit. Karena sifat terdistribusi EventBridge dan layanan target, dapat terjadi penundaan beberapa detik antara waktu aturan terjadwal dipicu dan waktu saat layanan target menjalankan sumber daya target.

Langkah:

- Langkah 1: Membuat fungsi Lambda
- Langkah 2: Buat Aturan
- Langkah 3: Verifikasi aturan
- Langkah 4: Konfirmasikan keberhasilan
- Langkah 5: Membersihkan sumber daya Anda

Langkah 1: Membuat fungsi Lambda

Buat fungsi Lambda untuk mencatat peristiwa yang dijadwalkan.

Untuk membuat fungsi Lambda

1. Buka AWS Lambda konsol tersebut di https://console.aws.amazon.com/lambda/.
2. Pilih Buat fungsi.
3. Pilih Tulis dari scratch.
4. Masukkan nama dan deskripsi untuk fungsi Lambda. Misalnya, beri nama fungsi tersebut `LogScheduledEvent`.
5. Tinggalkan sisa pilihan sebagai default dan pilih Buat fungsi.
6. Pada tab Kode dari halaman fungsi, klik dua kali index.js.
7. Ganti kode yang ada dengan kode berikut.

```
'use strict';
```

```
exports.handler = (event, context, callback) => {
    console.log('LogScheduledEvent');
    console.log('Received event:', JSON.stringify(event, null, 2));
    callback(null, 'Finished');
};
```

8. Pilih Deploy.

Langkah 2: Buat Aturan

Buat aturan untuk menjalankan fungsi Lambda yang Anda buat pada langkah 1 pada jadwal.

Anda dapat menggunakan konsol atau AWS CLI untuk membuat aturan. Untuk menggunakannya AWS CLI, Anda terlebih dahulu memberikan izin aturan untuk mengaktifkan fungsi Lambda Anda. Kemudian Anda dapat membuat aturan dan menambahkan fungsi Lambda sebagai target.

Untuk membuat aturan (konsol)

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan.

   Aturan tidak boleh memiliki nama yang sama dengan aturan lain di Wilayah yang sama dan di bus kejadian yang sama.
5. Untuk bus peristiwa, pilih bus kejadian yang ingin Anda kaitkan dengan aturan ini. Jika Anda ingin aturan ini cocok dengan kejadian yang berasal dari akun Anda, pilih bus kejadian default AWS. Saat layanan AWS di akun Anda menghasilkan kejadian, layanan tersebut akan selalu masuk ke bus kejadian default akun Anda.
6. Untuk jenis Aturan, pilih Jadwal.
7. Pilih Selanjutnya.
8. Untuk pola Jadwal, pilih Jadwal yang berjalan dengan tarif reguler, seperti setiap 10 menit. dan masuk**5** dan pilih Menit dari daftar menurun.
9. Pilih Selanjutnya.
10. Untuk jenis Target, pilih AWSlayanan.
11. Untuk Pilih target, pilih fungsi Lambda dari daftar drop-down.

12. Untuk Fungsi, pilih fungsi Lambda yang Anda buat. di bagian fungsi Lambda. Dalam contoh ini, pilih`LogScheduledEvent`.
13. Pilih Selanjutnya.
14. Pilih Selanjutnya.
15. Tinjau detail aturan dan pilih Buat aturan.

Untuk membuat aturan (AWS CLI)

1. Untuk membuat aturan yang berjalan pada jadwal, gunakan `put-rule` perintah.

```
aws events put-rule \
--name my-scheduled-rule \
--schedule-expression 'rate(5 minutes)'
```

Ketika aturan ini berjalan, itu membuat sebuah peristiwa dan kemudian mengirimkannya ke target. Berikut ini adalah contoh peristiwa.

```
{
    "version": "0",
    "id": "53dc4d37-cffa-4f76-80c9-8b7d4a4d2eaa",
    "detail-type": "Scheduled Event",
    "source": "aws.events",
    "account": "123456789012",
    "time": "2015-10-08T16:53:06Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:events:us-east-1:123456789012:rule/my-scheduled-rule"
    ],
    "detail": {}
}
```

2. Untuk memberikan izin EventBridge service principal (`events.amazonaws.com`) untuk menjalankan aturan, gunakan`add-permission` perintah.

```
aws lambda add-permission \
--function-name LogScheduledEvent \
--statement-id my-scheduled-event \
--action 'lambda:InvokeFunction' \
--principal events.amazonaws.com \
```

```
--source-arn arn:aws:events:us-east-1:123456789012:rule/my-scheduled-rule
```

3. Buat file `targets.json` dengan konten berikut.

```
[
  {
    "Id": "1",
    "Arn": "arn:aws:lambda:us-east-1:123456789012:function:LogScheduledEvent"
  }
]
```

4. Untuk menambahkan fungsi Lambda yang Anda buat di langkah 1 ke aturan, gunakan`put-targets` perintah.

```
aws events put-targets --rule my-scheduled-rule --targets file://targets.json
```

Langkah 3: Verifikasi aturan

Tunggu setidaknya lima menit setelah menyelesaikan langkah 2, lalu Anda dapat memverifikasi bahwa fungsi Lambda Anda dipanggil.

Melihat output dari fungsi Lambda Anda

1. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Di panel navigasi, pilih Log.
3. Pilih nama grup catatan untuk fungsi Lambda Anda (`/aws/lambda/function-name`).
4. Pilih nama aliran log untuk melihat data yang disediakan oleh fungsi untuk instans yang Anda luncurkan.

Langkah 4: Konfirmasikan keberhasilan

Jika melihat peristiwa Lambda di Logs (CloudWatch Lgs), Anda telah berhasil menyelesaikan tutorial ini. Jika peristiwa tersebut tidak ada di CloudWatch Logs (2), mulai pemecahan masalah dengan memverifikasi aturan berhasil dibuat dan, jika aturan terlihat benar, verifikasi kode fungsi Lambda Anda benar.

Langkah 5: Membersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankannya. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Untuk menghapus fungsi Lambda

1. Buka halaman Fungsi di konsol Lambda.
2. Pilih fungsi yang Anda buat.
3. Pilih Tindakan, Hapus.
4. Pilih Hapus.

Amazon EventBridge tutorial untuk mengintegrasikan dengan penyedia SaaS

EventBridge dapat bekerja secara langsung dengan aplikasi dan layanan mitra SaaS untuk mengirim dan menerimaacara. Tutorial berikut menunjukkan kepada Anda cara mengintegrasikan EventBridge dengan mitra SaaS.

Tutorial:

- Tutorial: Buat koneksi keDatadog sebagai tujuan API
- Tutorial: Membuat koneksi ke Salesforce sebagai tujuan API
- Tutorial: Buat koneksi keZendesk sebagai tujuan API

Tutorial: Buat koneksi keDatadog sebagai tujuan API

Anda dapat menggunakannya EventBridge untuk merutekan acara ke layanan pihak ketiga, seperti Datadog.

Dalam tutorial ini, Anda akan menggunakan EventBridge konsol untuk membuat koneksi keDatadog, tujuan API yang menunjuk keDatadog, dan aturan untuk merutekan peristiwa keDatadog.

Langkah:

- Prasyarat
- Langkah 1: Buat koneksi
- Langkah 2: Buat tujuan API
- Langkah 3: Buat aturan
- Langkah 4: Uji aturan
- Langkah 5: Membersihkan sumber daya Anda

Prasyarat

Untuk menyelesaikan tutorial ini, Anda memerlukan sumber daya berikut:

- Sebuah Datadogakun.
- KunciDatadog API.
- Bucket Amazon EventBridge Simple Storage Service (Amazon S3) yang diaktifkan.

Langkah 1: Buat koneksi

Untuk mengirim peristiwaDatadog, pertama-tama Anda harus membuat koneksi keDatadog API.

Untuk membuat koneksi

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih tujuan API.
3. Pilih tab Koneksi, lalu pilih Buat koneksi.
4. Masukkan nama dan deskripsi untuk koneksi. Misalnya, masukkan**`Datadog`** sebagai nama, dan**`Datadog API Connection`** sebagai deskripsi.

5. Untuk jenis Otorisasi, pilih kunci API.
6. Untuk nama kunci API, masukkan**`DD-API-KEY`**.
7. Untuk Nilai, tempelkan kunci APIDatadog rahasia Anda.
8. Pilih Create (Buat).

Langkah 2: Buat tujuan API

Sekarang setelah Anda membuat koneksi, selanjutnya Anda akan membuat tujuan API untuk digunakan sebagai target aturan.

Untuk membuat tujuan API

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih tujuan API.
3. Pilih Buat tujuan API.
4. Masukkan nama dan deskripsi untuk tujuan API. Misalnya, masukkan**`DatadogAD`** untuk nama, dan**`Datadog API Destination`** untuk deskripsi..
5. Untuk endpoint tujuan API, masukkan**`https://http-intake.logs.datadoghq.com/api/v2/logs`**.
6. Untuk metode HTTP, pilih POST.
7. Untuk batas tingkat pemanggilan, masukkan**`300`**.
8. Untuk Koneksi, pilih Gunakan koneksi yang ada dan pilih`Datadog` koneksi yang Anda buat di langkah 1.
9. Pilih Create (Buat).

Langkah 3: Buat aturan

Selanjutnya, Anda akan membuat aturan untuk mengirim peristiwaDatadog ketika objek Amazon S3 dibuat.

Untuk membuat tabel

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.

4. Masukkan nama dan deskripsi untuk aturan. Misalnya, masukkan**`DatadogRule`** nama, dan**`Rule to send events to Datadog for S3 object creation`** untuk deskripsi.
5. Untuk Event bus, pilih default.
6. Untuk jenis Aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.
8. Untuk Sumber acara, pilih Lainnya.
9. Untuk pola Event, masukkan yang berikut ini:

```
{
  "source": ["aws.s3"]
}
```

10. Pilih Selanjutnya.
11. Untuk jenis Target, pilih tujuanEventBridge API.
12. Untuk tujuan API, pilih Gunakan tujuan API yang ada, lalu pilih`DatadogAD` tujuan yang Anda buat di langkah 2.
13. Untuk peran Eksekusi, pilih Buat peran baru untuk sumber daya khusus ini.
14. Untuk Pengaturan tambahan, lakukan hal berikut:

    a. Untuk Konfigurasikan input target, pilih Transformator input dari daftar drop-down.

    b. Pilih Konfigurasikan transformator input

    c. untuk acara Contoh, masukkan berikut ini:

```
{
  "detail":[]
}
```

    d. Untuk transformator masukan Target lakukan hal berikut:

        i. Untuk Jalur Masukan, masukkan berikut ini:

```
{"detail":"$.detail"}
```

        ii. Untuk Template Masukan, masukkan berikut ini:

```
{"message": <detail>}
```

    e. Pilih Konfirmasi. .

15. Pilih Selanjutnya.
16. Pilih Selanjutnya.
17. Tinjau detail aturan dan pilih Buat aturan.

Langkah 4: Uji aturan

Untuk menguji aturan Anda, buat objek Amazon S3 dengan mengunggah file ke bucket EventBridge yang diaktifkan. Objek yang dibuat akan dicatat di konsolDatadog Log.

Langkah 5: Membersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankannya. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus EventBridge Koneksi

1. Buka halaman tujuan API EventBridge konsol.
2. Pilih tab Koneksi.
3. Pilih Koneksi yang Anda buat.
4. Pilih Delete (Hapus).
5. Masukkan nama koneksi dan pilih Hapus.

Untuk menghapus tujuan EventBridge API

1. Buka halaman tujuan API EventBridge konsol.
2. Pilih tujuan API yang Anda buat.
3. Pilih Delete (Hapus).
4. Masukkan nama tujuan API dan pilih Hapus.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan yang Anda buat.
3. Pilih Hapus.

4. Pilih Hapus.

Tutorial: Membuat koneksi ke Salesforce sebagai tujuan API

Anda dapat menggunakan EventBridge untuk merutekan acara ke layanan pihak ketiga, seperti Salesforce.

Dalam tutorial ini, Anda akan menggunakan EventBridge konsol untuk membuat koneksi keSalesforce, tujuan API yang menunjuk keSalesforce, dan aturan untuk merutekan peristiwaSalesforce.

Langkah:

- Prasyarat
- Langkah 1: Buat koneksi
- Langkah 2: Buat tujuan API
- Langkah 3: Buat aturan
- Langkah 4: Uji aturan
- Langkah 5: Membersihkan sumber daya Anda

Prasyarat

Untuk menyelesaikan tutorial ini, Anda memerlukan sumber daya berikut:

- Sebuah Salesforceakun.
- Aplikasi Salesforce yang terhubung.
- Token Salesforce keamanan.
- Acara platform Salesforce khusus.
- Bucket Amazon Simple Storage Service (Amazon S3) yang EventBridge diaktifkan.

Langkah 1: Buat koneksi

Untuk mengirim acara keSalesforce, Anda harus terlebih dahulu membuat koneksi ke Salesforce API.

Untuk membuat koneksi

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih tujuan API.
3. Pilih tab Connections, lalu pilih Create connection.

4. Masukkan nama dan deskripsi untuk koneksi. Misalnya, masukkan **`Salesforce`** sebagai nama, dan **`Salesforce API Connection`** sebagai deskripsi.
5. Untuk tipe Tujuan, pilih Partner dan Destinasi Mitra, pilih Salesforce dari daftar drop-down.
6. Untuk titik akhir Otorisasi, masukkan salah satu dari ini:

   - Jika Anda menggunakan org produksi, masukkan **`https://MyDomainName.my.salesforce.com./services/oauth2/token`**
   - Jika Anda menggunakan kotak pasir tanpa domain yang disempurnakan, masukkan **`https://MyDomainName--SandboxName.my. salesforce.com/services /oauth2/token`**
   - Jika Anda menggunakan kotak pasir dengan domain yang disempurnakan, masukkan **`https://MyDomainName-- SandboxName.sandbox.my.salesforce.com/services/oauth2/token`**
7. Untuk metode HTTP, pilih POST dari daftar drop-down.
8. Untuk ID Klien, masukkan ID klien dari aplikasi yang Salesforce terhubung.
9. Untuk rahasia Klien, masukkan rahasia klien dari aplikasi Anda yang Salesforce terhubung.
10. Untuk Parameter OAuth Http, masukkan pasangan kunci/nilai berikut:

Kunci	Nilai
grant_type	client_credentials

11. Pilih Buat.

Langkah 2: Buat tujuan API

Sekarang setelah Anda membuat koneksi, selanjutnya Anda akan membuat tujuan API untuk digunakan sebagai target aturan.

Untuk membuat API Destination

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih tujuan API.
3. Pilih Buat tujuan API.
4. Masukkan nama dan deskripsi untuk tujuan API. Misalnya, masukkan **`SalesforceAD`** untuk nama, dan **`Salesforce API Destination`** untuk deskripsi..

5. Untuk titik akhir tujuan API, masukkan **`https://MyDomainName.my.salesforce.com/services/data/v54.0/sobjects/MyEvent__e`** tempat MyEvent__e adalah acara platform tempat Anda ingin mengirim informasi.
6. Untuk metode HTTP, pilih POST dari daftar drop-down.
7. Untuk batas tingkat pemanggilan, masukkan. **`300`**
8. Untuk Koneksi, pilih Gunakan koneksi yang ada dan pilih `Salesforce` koneksi yang Anda buat di langkah 1.
9. Pilih Buat.

Langkah 3: Buat aturan

Selanjutnya, Anda akan membuat aturan untuk mengirim acara Salesforce saat objek Amazon S3 dibuat.

Untuk membuat tabel

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi qaturan. Misalnya, masukkan **`SalesforceRule`** untuk nama, dan **`Rule to send events to Salesforce for S3 object creation`** untuk deskripsi.
5. Untuk Bus peristiwa, pilih default.
6. Untuk Tipe aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.
8. Untuk sumber acara, pilih Lainnya.
9. Untuk pola Event, masukkan yang berikut ini:

```
{
  "source": ["aws.s3"]
}
```

10. Pilih Berikutnya.
11. Untuk jenis Target, pilih tujuan EventBridge API.
12. Untuk tujuan API, pilih Gunakan tujuan API yang ada, lalu pilih `SalesforceAD` tujuan yang Anda buat di langkah 2.

13. Untuk peran Eksekusi, pilih Buat peran baru untuk sumber daya khusus ini.
14. Untuk pengaturan tambahan, lakukan hal berikut:
    a. Untuk Konfigurasi input target, pilih Input transformator dari daftar drop-down.
    b. Pilih Konfigurasi transformator input
    c. untuk Contoh acara, masukkan yang berikut ini:

```
{
  "detail":[]
}
```

    d. Untuk transformator input Target lakukan hal berikut:
        i. Untuk Jalur Input, masukkan yang berikut ini:

```
{"detail":"$.detail"}
```

        ii. Untuk Template Input, masukkan yang berikut ini:

```
{"message": <detail>}
```

    e. Pilih Konfirmasi. .
15. Pilih Selanjutnya.
16. Pilih Selanjutnya.
17. Tinjau detail aturan dan pilih Buat aturan.

Langkah 4: Uji aturan

Untuk menguji aturan Anda, buat objek Amazon S3 dengan mengunggah file ke bucket yang diaktifkan. EventBridge Informasi tentang objek yang dibuat akan dikirim ke acara Salesforce platform.

Langkah 5: Membersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankannya. Dengan menghapus AWS sumber daya yang tidak lagi Anda gunakan, Anda mencegah tagihan yang tidak perlu ke AWS akun Anda.

Untuk menghapus EventBridge Koneksi

1. Buka halaman tujuan API EventBridge konsol.
2. Pilih tab Koneksi.
3. Pilih Koneksi yang Anda buat.
4. Pilih Hapus.
5. Masukkan nama koneksi dan pilih Hapus.

Untuk menghapus tujuan EventBridge API

1. Buka halaman tujuan API EventBridge konsol.
2. Pilih tujuan API yang Anda buat.
3. Pilih Hapus.
4. Masukkan nama tujuan API dan pilih Hapus.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan yang Anda buat.
3. Pilih Hapus.
4. Pilih Hapus.

Tutorial: Buat koneksi keZendesk sebagai tujuan API

Anda dapat menggunakan EventBridge untuk rute acara ke layanan pihak ketiga seperti Zendesk.

Dalam tutorial ini, Anda akan menggunakan EventBridge konsol untuk membuat koneksi keZendesk, tujuan API yang menunjuk keZendesk, dan aturan untuk merutekan peristiwa keZendesk.

Langkah:

- Prasyarat
- Langkah 1: Buat koneksi
- Langkah 2: Buat tujuan API
- Langkah 3: Buat aturan
- Langkah 4: Uji aturan
- Langkah 5: Membersihkan sumber daya Anda

Prasyarat

Untuk menyelesaikan tutorial ini, Anda memerlukan sumber daya berikut:

- Sebuah Zendeskakun.
- Bucket EventBridge Amazon Simple Storage Service (Amazon S3).

Langkah 1: Buat koneksi

Untuk mengirim peristiwaZendesk, pertama-tama Anda harus membuat koneksi keZendesk API.

Untuk membuat koneksi

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih tujuan API.
3. Pilih tab Koneksi, lalu pilih Buat koneksi.
4. Masukkan nama dan deskripsi untuk koneksi. Misalnya, masukkan**`Zendesk`** nama, dan**`Connection to Zendesk API`** untuk deskripsi.
5. Untuk jenis Otorisasi, pilih Dasar (Nama Pengguna/Kata Sandi).
6. Untuk Nama Pengguna, masukkanZendesk nama pengguna Anda.

7. Untuk Kata Sandi, masukkanZendesk kata sandi Anda.
8. Pilih Create (Buat).

Langkah 2: Buat tujuan API

Sekarang setelah Anda membuat koneksi, Anda selanjutnya akan membuat tujuan API untuk digunakan sebagai target aturan.

Untuk membuat Tujuan API

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih tujuan API.
3. Pilih Buat tujuan API.
4. Masukkan nama dan deskripsi untuk tujuan API. Misalnya, masukkan**`ZendeskAD`** nama, dan**`Zendesk API destination`** untuk deskripsi.
5. Untuk endpoint tujuan API, masukkan**`https://`*`your-subdomain`*`.zendesk.com/api/v2/tickets.json`**, di *`mana subdomain`* Anda adalah subdomain yang terkait denganZendesk akun Anda.
6. Untuk metode HTTP, pilih POST.
7. Untuk batas tingkat pemanggilan, masukkan**`10`**.
8. Untuk Koneksi, pilih Gunakan koneksi yang ada dan pilih`Zendesk` koneksi yang Anda buat di langkah 1.
9. Pilih Create (Buat).

Langkah 3: Buat aturan

Selanjutnya, buat aturan untuk mengirim peristiwaZendesk ketika objek Amazon S3 dibuat.

Untuk membuat tabel

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di panel navigasi, pilih Aturan.
3. Pilih Buat aturan.
4. Masukkan nama dan deskripsi untuk aturan. Misalnya, masukkan**`ZendeskRule`** nama, dan**`Rule to send events to Zendesk when S3 objects are created`** untuk deskripsi.

5. Untuk Event bus, pilih default.
6. Untuk jenis Aturan, pilih Aturan dengan pola peristiwa.
7. Pilih Selanjutnya.
8. Untuk Sumber acara, pilih Lainnya.
9. Untuk pola Event, masukkan yang berikut ini:

```
{
  "source": ["aws.s3"]
}
```

10. Pilih Selanjutnya.
11. Untuk jenis Target, pilih tujuanEventBridge API.
12. Untuk tujuan API, pilih Gunakan tujuan API yang ada, lalu pilih`ZendeskAD` tujuan yang Anda buat di langkah 2.
13. Untuk peran Eksekusi, pilih Buat peran baru untuk sumber daya khusus ini.
14. Untuk Pengaturan tambahan, lakukan hal berikut:
    a. Untuk Konfigurasikan input target, pilih Transformator input dari daftar drop-down.
    b. Pilih Konfigurasi transformator input
    c. untuk acara Contoh, masukkan berikut ini:

```
{
  "detail":[]
}
```

    d. Untuk transformator masukan Target lakukan hal berikut:
        i. Untuk Jalur input, masukkan berikut ini:

```
{"detail":"$.detail"}
```

        ii. Untuk Template input, masukkan berikut ini:

```
{"message": <detail>}
```

    e. Pilih Konfirmasi. .
15. Pilih Selanjutnya.

16. Pilih Selanjutnya.
17. Tinjau detail aturan dan pilih Buat aturan.

Langkah 4: Uji aturan

Untuk menguji aturan Anda, buat objek Amazon S3 dengan mengunggah file ke bucket EventBridge yang diaktifkan. Ketika acara cocok dengan aturan, EventBridge akan memanggil ZendeskCreate Ticket API. Tiket baru akan muncul diZendesk dasbor.

Langkah 5: Membersihkan sumber daya Anda

Sekarang Anda dapat menghapus sumber daya yang Anda buat untuk tutorial ini, kecuali Anda ingin mempertahankannya. Dengan menghapus AWS sumber daya AWS yang tidak lagi Anda gunakan, Anda mencegah biaya yang tidak perlu di AWS akun Anda.

Untuk menghapus EventBridge Koneksi

1. Buka halaman tujuan API EventBridge konsol.
2. Pilih tab Koneksi.
3. Pilih Koneksi yang Anda buat.
4. Pilih Delete (Hapus).
5. Masukkan nama koneksi dan pilih Hapus.

Untuk menghapus tujuan EventBridge API

1. Buka halaman tujuan API EventBridge konsol.
2. Pilih tujuan API yang Anda buat.
3. Pilih Delete (Hapus).
4. Masukkan nama tujuan API dan pilih Hapus.

Untuk menghapus EventBridge aturan

1. Buka halaman Aturan EventBridge konsol.
2. Pilih aturan (s) yang Anda buat.
3. Pilih Hapus.

4. Pilih Hapus.

Menggunakan EventBridge dengan AWS SDK

AWS kit pengembangan perangkat lunak (SDK) tersedia untuk banyak bahasa pemrograman populer. Setiap SDK menyediakan API, contoh kode, dan dokumentasi yang memudahkan developer untuk membangun aplikasi dalam bahasa pilihan mereka.

Dokumentasi SDK	Contoh kode
AWS SDK for C++	AWS SDK for C++ contoh kode
AWS SDK for Go	AWS SDK for Go contoh kode
AWS SDK for Java	AWS SDK for Java contoh kode
AWS SDK for JavaScript	AWS SDK for JavaScript contoh kode
AWS SDK for Kotlin	AWS SDK for Kotlin contoh kode
AWS SDK for .NET	AWS SDK for .NET contoh kode
AWS SDK for PHP	AWS SDK for PHP contoh kode
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) contoh kode
AWS SDK for Ruby	AWS SDK for Ruby contoh kode
AWS SDK for Rust	AWS SDK for Rust contoh kode
AWS SDK untuk SAP ABAP	AWS SDK untuk SAP ABAP contoh kode
AWS SDK for Swift	AWS SDK for Swift contoh kode

Untuk contoh khusus untuk EventBridge, lihatContoh kode untuk EventBridge menggunakan AWS SDK.

Ketersediaan contoh

Tidak menemukan yang Anda cari? Minta contoh kode menggunakan tautan Berikan umpan balik di bagian bawah halaman ini.

Contoh kode untuk EventBridge menggunakan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan EventBridge kit pengembangan AWS perangkat lunak (SDK).

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Meskipun tindakan menunjukkan cara memanggil setiap fungsi layanan, Anda dapat melihat tindakan dalam konteks pada skenario yang terkait dan contoh lintas layanan.

Skenario adalah contoh kode yang menunjukkan cara untuk menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan yang sama.

Contoh lintas layanan adalah contoh aplikasi yang bekerja di beberapa Layanan AWS.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga berisi informasi tentang cara memulai dan detail tentang versi SDK sebelumnya.

Memulai

Halo EventBridge

Contoh kode berikut menunjukkan cara untuk mulai menggunakan EventBridge.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
using Amazon.EventBridge;
using Amazon.EventBridge.Model;

namespace EventBridgeActions;
```

```
public static class HelloEventBridge
{
    static async Task Main(string[] args)
    {
        var eventBridgeClient = new AmazonEventBridgeClient();

        Console.WriteLine($"Hello Amazon EventBridge! Following are some of your
 EventBuses:");
        Console.WriteLine();

        // You can use await and any of the async methods to get a response.
        // Let's get the first five event buses.
        var response = await eventBridgeClient.ListEventBusesAsync(
            new ListEventBusesRequest()
            {
                Limit = 5
            });

        foreach (var eventBus in response.EventBuses)
        {
            Console.WriteLine($"\tEventBus: {eventBus.Name}");
            Console.WriteLine($"\tArn: {eventBus.Arn}");
            Console.WriteLine($"\tPolicy: {eventBus.Policy}");
            Console.WriteLine();
        }
    }
}
```

- Untuk detail API, lihat ListEventBusesdi Referensi AWS SDK for .NET API.

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
/**
```

```
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 */
public class HelloEventBridge {
    public static void main(String[] args) {
        Region region = Region.US_WEST_2;
        EventBridgeClient eventBrClient = EventBridgeClient.builder()
                .region(region)
                .build();

        listBuses(eventBrClient);
        eventBrClient.close();
    }

    public static void listBuses(EventBridgeClient eventBrClient) {
        try {
            ListEventBusesRequest busesRequest = ListEventBusesRequest.builder()
                    .limit(10)
                    .build();

            ListEventBusesResponse response =
 eventBrClient.listEventBuses(busesRequest);
            List<EventBus> buses = response.eventBuses();
            for (EventBus bus : buses) {
                System.out.println("The name of the event bus is: " +
 bus.name());
                System.out.println("The ARN of the event bus is: " + bus.arn());
            }

        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat ListEventBusesdi Referensi AWS SDK for Java 2.x API.

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import aws.sdk.kotlin.services.eventbridge.EventBridgeClient
import aws.sdk.kotlin.services.eventbridge.model.ListEventBusesRequest
import aws.sdk.kotlin.services.eventbridge.model.ListEventBusesResponse

suspend fun main() {
    listBusesHello()
}

suspend fun listBusesHello() {
    val request = ListEventBusesRequest {
        limit = 10
    }

    EventBridgeClient { region = "us-west-2" }.use { eventBrClient ->
        val response: ListEventBusesResponse =
 eventBrClient.listEventBuses(request)
        response.eventBuses?.forEach { bus ->
            println("The name of the event bus is ${bus.name}")
            println("The ARN of the event bus is ${bus.arn}")
        }
    }
}
```

- Untuk detail API, lihat ListEventBusesdi AWS SDK untuk referensi API Kotlin.

Contoh kode

- Tindakan untuk EventBridge menggunakan AWS SDK
  - Gunakan DeleteRule dengan AWS SDK atau alat baris perintah
  - Gunakan DescribeRule dengan AWS SDK atau alat baris perintah

- Gunakan DisableRule dengan AWS SDK atau alat baris perintah
- Gunakan EnableRule dengan AWS SDK atau alat baris perintah
- Gunakan ListRuleNamesByTarget dengan AWS SDK atau alat baris perintah
- Gunakan ListRules dengan AWS SDK atau alat baris perintah
- Gunakan ListTargetsByRule dengan AWS SDK atau alat baris perintah
- Gunakan PutEvents dengan AWS SDK atau alat baris perintah
- Gunakan PutRule dengan AWS SDK atau alat baris perintah
- Gunakan PutTargets dengan AWS SDK atau alat baris perintah
- Gunakan RemoveTargets dengan AWS SDK atau alat baris perintah
- Skenario untuk EventBridge menggunakan AWS SDK
  - Membuat dan memicu aturan di Amazon EventBridge menggunakan AWS SDK
  - Memulai EventBridge aturan dan target menggunakan AWS SDK
- Contoh lintas layanan untuk EventBridge menggunakan AWS SDK
  - Menggunakan peristiwa terjadwal untuk menginvokasi fungsi Lambda

Tindakan untuk EventBridge menggunakan AWS SDK

Contoh kode berikut menunjukkan cara melakukan EventBridge tindakan individual dengan AWS SDK. Kutipan ini memanggil EventBridge API dan merupakan kutipan kode dari program yang lebih besar yang harus dijalankan dalam konteks. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkapnya, lihat Referensi Amazon EventBridge API.

Contoh-contoh

- Gunakan DeleteRule dengan AWS SDK atau alat baris perintah
- Gunakan DescribeRule dengan AWS SDK atau alat baris perintah
- Gunakan DisableRule dengan AWS SDK atau alat baris perintah
- Gunakan EnableRule dengan AWS SDK atau alat baris perintah
- Gunakan ListRuleNamesByTarget dengan AWS SDK atau alat baris perintah
- Gunakan ListRules dengan AWS SDK atau alat baris perintah

- Gunakan ListTargetsByRule dengan AWS SDK atau alat baris perintah
- Gunakan PutEvents dengan AWS SDK atau alat baris perintah
- Gunakan PutRule dengan AWS SDK atau alat baris perintah
- Gunakan PutTargets dengan AWS SDK atau alat baris perintah
- Gunakan RemoveTargets dengan AWS SDK atau alat baris perintah

Gunakan **`DeleteRule`** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan`DeleteRule`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- Memulai dengan aturan dan target

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Hapus aturan dengan namanya.

```
/// <summary>
/// Delete an event rule by name.
/// </summary>
/// <param name="ruleName">The name of the event rule.</param>
/// <returns>True if successful.</returns>
public async Task<bool> DeleteRuleByName(string ruleName)
{
    var response = await _amazonEventBridge.DeleteRuleAsync(
        new DeleteRuleRequest()
        {
            Name = ruleName
```

```
            });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat DeleteRuledi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk menghapus aturan CloudWatch Acara

Contoh ini menghapus aturan bernama InstanceStateChanges EC2:

```
aws events delete-rule --name "EC2InstanceStateChanges"
```

- Untuk detail API, lihat DeleteRuledi Referensi AWS CLI Perintah.

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
   public static void deleteRuleByName(EventBridgeClient eventBrClient, String
ruleName) {
       DeleteRuleRequest ruleRequest = DeleteRuleRequest.builder()
               .name(ruleName)
               .build();

       eventBrClient.deleteRule(ruleRequest);
       System.out.println("Successfully deleted the rule");
   }
```

- Untuk detail API, lihat DeleteRuledi Referensi AWS SDK for Java 2.x API.

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun deleteRuleByName(ruleName: String?) {
    val ruleRequest = DeleteRuleRequest {
        name = ruleName
    }
    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.deleteRule(ruleRequest)
        println("Successfully deleted the rule")
    }
}
```

- Untuk detail API, lihat DeleteRuledi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **`DescribeRule`** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan`DescribeRule`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- Memulai dengan aturan dan target

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan status aturan menggunakan deskripsi aturan.

```
    /// <summary>
    /// Get the state for a rule by the rule name.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <param name="eventBusName">The optional name of the event bus. If empty,
uses the default event bus.</param>
    /// <returns>The state of the rule.</returns>
    public async Task<RuleState> GetRuleStateByRuleName(string ruleName, string?
eventBusName = null)
    {
        var ruleResponse = await _amazonEventBridge.DescribeRuleAsync(
            new DescribeRuleRequest()
            {
                Name = ruleName,
                EventBusName = eventBusName
            });
        return ruleResponse.State;
    }
```

- Untuk detail API, lihat DescribeRuledi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk menampilkan informasi tentang aturan CloudWatch Acara

Contoh ini menampilkan informasi tentang aturan bernama DailyLambdaFunction:

```
aws events describe-rule --name "DailyLambdaFunction"
```

- Untuk detail API, lihat DescribeRuledi Referensi AWS CLI Perintah.

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
   public static void checkRule(EventBridgeClient eventBrClient, String
eventRuleName) {
       try {
           DescribeRuleRequest ruleRequest = DescribeRuleRequest.builder()
                   .name(eventRuleName)
                   .build();

           DescribeRuleResponse response =
eventBrClient.describeRule(ruleRequest);
           System.out.println("The state of the rule is " +
response.stateAsString());

       } catch (EventBridgeException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Untuk detail API, lihat DescribeRuledi Referensi AWS SDK for Java 2.x API.

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun checkRule(eventRuleName: String?) {
    val ruleRequest = DescribeRuleRequest {
        name = eventRuleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.describeRule(ruleRequest)
        println("The state of the rule is $response")
    }
}
```

- Untuk detail API, lihat DescribeRuledi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **`DisableRule`** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan`DisableRule`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- Memulai dengan aturan dan target

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Nonaktifkan aturan dengan nama aturannya.

```
/// <summary>
/// Disable a particular rule on an event bus.
/// </summary
/// <param name="ruleName">The name of the rule.</param>
/// <returns>True if successful.</returns>
public async Task<bool> DisableRuleByName(string ruleName)
{
    var ruleResponse = await _amazonEventBridge.DisableRuleAsync(
        new DisableRuleRequest()
        {
            Name = ruleName
        });
    return ruleResponse.HttpStatusCode == HttpStatusCode.OK;
}
```

- Untuk detail API, lihat DisableRuledi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk menonaktifkan aturan CloudWatch Acara

Contoh ini menonaktifkan aturan bernama DailyLambdaFunction. Aturan tidak dihapus:

```
aws events disable-rule --name "DailyLambdaFunction"
```

- Untuk detail API, lihat DisableRuledi Referensi AWS CLI Perintah.

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Nonaktifkan aturan dengan menggunakan nama aturannya.

```
   public static void changeRuleState(EventBridgeClient eventBrClient, String
eventRuleName, Boolean isEnabled) {
       try {
           if (!isEnabled) {
               System.out.println("Disabling the rule: " + eventRuleName);
               DisableRuleRequest ruleRequest = DisableRuleRequest.builder()
                       .name(eventRuleName)
                       .build();

               eventBrClient.disableRule(ruleRequest);
           } else {
               System.out.println("Enabling the rule: " + eventRuleName);
               EnableRuleRequest ruleRequest = EnableRuleRequest.builder()
                       .name(eventRuleName)
                       .build();
               eventBrClient.enableRule(ruleRequest);
           }

       } catch (EventBridgeException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Untuk detail API, lihat DisableRuledi Referensi AWS SDK for Java 2.x API.

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun changeRuleState(eventRuleName: String, isEnabled: Boolean?) {
    if (!isEnabled!!) {
        println("Disabling the rule: $eventRuleName")
        val ruleRequest = DisableRuleRequest {
            name = eventRuleName
        }
        EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
            eventBrClient.disableRule(ruleRequest)
        }
    } else {
        println("Enabling the rule: $eventRuleName")
        val ruleRequest = EnableRuleRequest {
            name = eventRuleName
        }
        EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
            eventBrClient.enableRule(ruleRequest)
        }
    }
}
```

- Untuk detail API, lihat DisableRuledi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **`EnableRule`** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan`EnableRule`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- Memulai dengan aturan dan target

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Aktifkan aturan dengan nama aturannya.

```
/// <summary>
/// Enable a particular rule on an event bus.
/// </summary>
/// <param name="ruleName">The name of the rule.</param>
/// <returns>True if successful.</returns>
public async Task<bool> EnableRuleByName(string ruleName)
{
    var ruleResponse = await _amazonEventBridge.EnableRuleAsync(
        new EnableRuleRequest()
        {
            Name = ruleName
        });
    return ruleResponse.HttpStatusCode == HttpStatusCode.OK;
}
```

- Untuk detail API, lihat EnableRuledi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk mengaktifkan aturan CloudWatch Acara

Contoh ini memungkinkan aturan bernama DailyLambdaFunction, yang sebelumnya telah dinonaktifkan:

```
aws events enable-rule --name "DailyLambdaFunction"
```

- Untuk detail API, lihat EnableRuledi Referensi AWS CLI Perintah.

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Aktifkan aturan dengan menggunakan nama aturannya.

```
   public static void changeRuleState(EventBridgeClient eventBrClient, String
eventRuleName, Boolean isEnabled) {
       try {
           if (!isEnabled) {
               System.out.println("Disabling the rule: " + eventRuleName);
               DisableRuleRequest ruleRequest = DisableRuleRequest.builder()
                       .name(eventRuleName)
                       .build();

               eventBrClient.disableRule(ruleRequest);
           } else {
               System.out.println("Enabling the rule: " + eventRuleName);
               EnableRuleRequest ruleRequest = EnableRuleRequest.builder()
                       .name(eventRuleName)
                       .build();
               eventBrClient.enableRule(ruleRequest);
           }

       } catch (EventBridgeException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
```

```
    }
```

- Untuk detail API, lihat EnableRuledi Referensi AWS SDK for Java 2.x API.

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun changeRuleState(eventRuleName: String, isEnabled: Boolean?) {
    if (!isEnabled!!) {
        println("Disabling the rule: $eventRuleName")
        val ruleRequest = DisableRuleRequest {
            name = eventRuleName
        }
        EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
            eventBrClient.disableRule(ruleRequest)
        }
    } else {
        println("Enabling the rule: $eventRuleName")
        val ruleRequest = EnableRuleRequest {
            name = eventRuleName
        }
        EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
            eventBrClient.enableRule(ruleRequest)
        }
    }
}
```

- Untuk detail API, lihat EnableRuledi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan `ListRuleNamesByTarget` dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan`ListRuleNamesByTarget`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- Memulai dengan aturan dan target

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Daftar semua nama aturan menggunakan target.

```
/// <summary>
/// List names of all rules matching a target.
/// </summary>
/// <param name="targetArn">The ARN of the target.</param>
/// <returns>The list of rule names.</returns>
public async Task<List<string>> ListAllRuleNamesByTarget(string targetArn)
{
    var results = new List<string>();
    var request = new ListRuleNamesByTargetRequest()
    {
        TargetArn = targetArn
    };
    ListRuleNamesByTargetResponse response;
    do
    {
```

```
            response = await
_amazonEventBridge.ListRuleNamesByTargetAsync(request);
            results.AddRange(response.RuleNames);
            request.NextToken = response.NextToken;

        } while (response.NextToken is not null);

        return results;
    }
```

- Untuk detail API, lihat ListRuleNamesByTargetdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk menampilkan semua aturan yang memiliki target tertentu

Contoh ini menampilkan semua aturan yang memiliki fungsi Lambda bernama "MyFunctionName" sebagai target:

```
aws events list-rule-names-by-target --target-arn "arn:aws:lambda:us-
east-1:123456789012:function:MyFunctionName"
```

- Untuk detail API, lihat ListRuleNamesByTargetdi Referensi AWS CLI Perintah.

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar semua nama aturan dengan menggunakan target.

```
    public static void listTargetRules(EventBridgeClient eventBrClient, String
topicArn) {
```

```
        ListRuleNamesByTargetRequest ruleNamesByTargetRequest =
 ListRuleNamesByTargetRequest.builder()
                .targetArn(topicArn)
                .build();

        ListRuleNamesByTargetResponse response =
 eventBrClient.listRuleNamesByTarget(ruleNamesByTargetRequest);
        List<String> rules = response.ruleNames();
        for (String rule : rules) {
            System.out.println("The rule name is " + rule);
        }
    }
```

- Untuk detail API, lihat ListRuleNamesByTargetdi Referensi AWS SDK for Java 2.x API.

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun listTargetRules(topicArnVal: String?) {
    val ruleNamesByTargetRequest = ListRuleNamesByTargetRequest {
        targetArn = topicArnVal
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response =
 eventBrClient.listRuleNamesByTarget(ruleNamesByTargetRequest)
        response.ruleNames?.forEach { rule ->
            println("The rule name is $rule")
        }
    }
}
```

- Untuk detail API, lihat ListRuleNamesByTargetdi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan `ListRules` dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan`ListRules`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- Memulai dengan aturan dan target

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Daftar semua aturan untuk bus acara.

```
    /// <summary>
    /// List the rules on an event bus.
    /// </summary>
    /// <param name="eventBusArn">The optional ARN of the event bus. If empty,
uses the default event bus.</param>
    /// <returns>The list of rules.</returns>
    public async Task<List<Rule>> ListAllRulesForEventBus(string? eventBusArn =
null)
    {
        var results = new List<Rule>();
        var request = new ListRulesRequest()
        {
            EventBusName = eventBusArn
```

```
        };
        // Get all of the pages of rules.
        ListRulesResponse response;
        do
        {
            response = await _amazonEventBridge.ListRulesAsync(request);
            results.AddRange(response.Rules);
            request.NextToken = response.NextToken;

        } while (response.NextToken is not null);

        return results;
    }
```

- Untuk detail API, lihat ListRulesdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk menampilkan daftar semua aturan CloudWatch Acara

Contoh ini menampilkan semua aturan CloudWatch Acara di wilayah:

```
aws events list-rules
```

Untuk menampilkan daftar aturan CloudWatch Peristiwa yang dimulai dengan string tertentu.

Contoh ini menampilkan semua aturan CloudWatch Acara di wilayah yang memiliki nama yang dimulai dengan “Harian”:

```
aws events list-rules --name-prefix "Daily"
```

- Untuk detail API, lihat ListRulesdi Referensi AWS CLI Perintah.

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Aktifkan aturan dengan menggunakan nama aturannya.

```
    public static void listRules(EventBridgeClient eventBrClient) {
        try {
            ListRulesRequest rulesRequest = ListRulesRequest.builder()
                    .eventBusName("default")
                    .limit(10)
                    .build();

            ListRulesResponse response = eventBrClient.listRules(rulesRequest);
            List<Rule> rules = response.rules();
            for (Rule rule : rules) {
                System.out.println("The rule name is : " + rule.name());
                System.out.println("The rule description is : " +
rule.description());
                System.out.println("The rule state is : " +
rule.stateAsString());
            }

        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Untuk detail API, lihat ListRulesdi Referensi AWS SDK for Java 2.x API.

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun listRules() {
    val rulesRequest = ListRulesRequest {
        eventBusName = "default"
        limit = 10
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.listRules(rulesRequest)
        response.rules?.forEach { rule ->
            println("The rule name is ${rule.name}")
            println("The rule ARN is ${rule.arn}")
        }
    }
}
```

- Untuk detail API, lihat ListRulesdi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan `ListTargetsByRule` dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan`ListTargetsByRule`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- Memulai dengan aturan dan target

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar semua target untuk aturan menggunakan nama aturan.

```
/// <summary>
/// List all of the targets matching a rule by name.
/// </summary>
/// <param name="ruleName">The name of the rule.</param>
/// <returns>The list of targets.</returns>
public async Task<List<Target>> ListAllTargetsOnRule(string ruleName)
{
    var results = new List<Target>();
    var request = new ListTargetsByRuleRequest()
    {
        Rule = ruleName
    };
    ListTargetsByRuleResponse response;
    do
    {
        response = await _amazonEventBridge.ListTargetsByRuleAsync(request);
        results.AddRange(response.Targets);
        request.NextToken = response.NextToken;

    } while (response.NextToken is not null);

    return results;
}
```

- Untuk detail API, lihat ListTargetsByRuledi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk menampilkan semua target untuk aturan CloudWatch Acara

Contoh ini menampilkan semua target dari aturan bernama DailyLambdaFunction:

```
aws events list-targets-by-rule --rule  "DailyLambdaFunction"
```

- Untuk detail API, lihat ListTargetsByRuledi Referensi AWS CLI Perintah.

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar semua target untuk aturan dengan menggunakan nama aturan.

```
   public static void listTargets(EventBridgeClient eventBrClient, String
ruleName) {
       ListTargetsByRuleRequest ruleRequest = ListTargetsByRuleRequest.builder()
               .rule(ruleName)
               .build();

       ListTargetsByRuleResponse res =
eventBrClient.listTargetsByRule(ruleRequest);
       List<Target> targetsList = res.targets();
       for (Target target: targetsList) {
           System.out.println("Target ARN: "+target.arn());
       }
   }
```

- Untuk detail API, lihat ListTargetsByRuledi Referensi AWS SDK for Java 2.x API.

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun listTargets(ruleName: String?) {
    val ruleRequest = ListTargetsByRuleRequest {
        rule = ruleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.listTargetsByRule(ruleRequest)
        response.targets?.forEach { target ->
            println("Target ARN: ${target.arn}")
        }
    }
}
```

- Untuk detail API, lihat ListTargetsByRuledi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **`PutEvents`** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan`PutEvents`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat dan picu aturan
- Memulai dengan aturan dan target

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Kirim acara yang cocok dengan pola kustom untuk aturan.

```
    /// <summary>
    /// Add an event to the event bus that includes an email, message, and time.
    /// </summary>
    /// <param name="email">The email to use in the event detail of the custom
event.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> PutCustomEmailEvent(string email)
    {
        var eventDetail = new
        {
            UserEmail = email,
            Message = "This event was generated by example code.",
            UtcTime = DateTime.UtcNow.ToString("g")
        };
        var response = await _amazonEventBridge.PutEventsAsync(
            new PutEventsRequest()
            {
                Entries = new List<PutEventsRequestEntry>()
                {
                    new PutEventsRequestEntry()
                    {
                        Source = "ExampleSource",
                        Detail = JsonSerializer.Serialize(eventDetail),
                        DetailType = "ExampleType"
                    }
                }
            });

        return response.FailedEntryCount == 0;
    }
```

- Untuk detail API, lihat PutEventsdi Referensi AWS SDK for .NET API.

C++

SDK for C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Sertakan file-file yang diperlukan.

```
#include <aws/core/Aws.h>
#include <aws/events/EventBridgeClient.h>
#include <aws/events/model/PutEventsRequest.h>
#include <aws/events/model/PutEventsResult.h>
#include <aws/core/utils/Outcome.h>
#include <iostream>
```

Kirim acara.

```
        Aws::CloudWatchEvents::EventBridgeClient cwe;

        Aws::CloudWatchEvents::Model::PutEventsRequestEntry event_entry;
        event_entry.SetDetail(MakeDetails(event_key, event_value));
        event_entry.SetDetailType("sampleSubmitted");
        event_entry.AddResources(resource_arn);
        event_entry.SetSource("aws-sdk-cpp-cloudwatch-example");

        Aws::CloudWatchEvents::Model::PutEventsRequest request;
        request.AddEntries(event_entry);

        auto outcome = cwe.PutEvents(request);
        if (!outcome.IsSuccess())
        {
            std::cout << "Failed to post CloudWatch event: " <<
```

```
                outcome.GetError().GetMessage() << std::endl;
        }
        else
        {
            std::cout << "Successfully posted CloudWatch event" << std::endl;
        }
```

- Untuk detail API, lihat PutEventsdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Untuk mengirim acara khusus ke CloudWatch Acara

Contoh ini mengirimkan acara khusus ke CloudWatch Acara. Acara ini terkandung dalam file putevents.json:

```
aws events put-events --entries file://putevents.json
```

Berikut adalah isi dari file putevents.json:

```
[
  {
    "Source": "com.mycompany.myapp",
    "Detail": "{ \"key1\": \"value1\", \"key2\": \"value2\" }",
    "Resources": [
      "resource1",
      "resource2"
    ],
    "DetailType": "myDetailType"
  },
  {
    "Source": "com.mycompany.myapp",
    "Detail": "{ \"key1\": \"value3\", \"key2\": \"value4\" }",
    "Resources": [
      "resource1",
      "resource2"
    ],
    "DetailType": "myDetailType"
   }
```

```
]
```

- Untuk detail API, lihat PutEventsdi Referensi AWS CLI Perintah.

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
   public static void triggerCustomRule(EventBridgeClient eventBrClient, String
email) {
       String json = "{" +
               "\"UserEmail\": \"" + email + "\"," +
               "\"Message\": \"This event was generated by example code.\"," +
               "\"UtcTime\": \"Now.\"" +
               "}";

       PutEventsRequestEntry entry = PutEventsRequestEntry.builder()
               .source("ExampleSource")
               .detail(json)
               .detailType("ExampleType")
               .build();

       PutEventsRequest eventsRequest = PutEventsRequest.builder()
               .entries(entry)
               .build();

       eventBrClient.putEvents(eventsRequest);
   }
```

- Untuk detail API, lihat PutEventsdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Mengimpor modul SDK dan klien dan memanggil API.

```
import {
  EventBridgeClient,
  PutEventsCommand,
} from "@aws-sdk/client-eventbridge";

export const putEvents = async (
  source = "eventbridge.integration.test",
  detailType = "greeting",
  resources = [],
) => {
  const client = new EventBridgeClient({});

  const response = await client.send(
    new PutEventsCommand({
      Entries: [
        {
          Detail: JSON.stringify({ greeting: "Hello there." }),
          DetailType: detailType,
          Resources: resources,
          Source: source,
        },
      ],
    }),
  );

  console.log("PutEvents response:");
  console.log(response);
  // PutEvents response:
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
```

```
  //     requestId: '3d0df73d-dcea-4a23-ae0d-f5556a3ac109',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   Entries: [ { EventId: '51620841-5af4-6402-d9bc-b77734991eb5' } ],
  //   FailedEntryCount: 0
  // }

  return response;
};
```

- Untuk detail API, lihat PutEventsdi Referensi AWS SDK for JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create CloudWatchEvents service object
var ebevents = new AWS.EventBridge({ apiVersion: "2015-10-07" });

var params = {
  Entries: [
    {
      Detail: '{ "key1": "value1", "key2": "value2" }',
      DetailType: "appRequestSubmitted",
      Resources: ["RESOURCE_ARN"],
      Source: "com.company.app",
    },
  ],
};
```

```
ebevents.putEvents(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.Entries);
  }
});
```

- Untuk detail API, lihat PutEventsdi Referensi AWS SDK for JavaScript API.

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun triggerCustomRule(email: String) {
    val json = "{" +
        "\"UserEmail\": \"" + email + "\"," +
        "\"Message\": \"This event was generated by example code.\"" +
        "\"UtcTime\": \"Now.\"" +
        "}"

    val entry = PutEventsRequestEntry {
        source = "ExampleSource"
        detail = json
        detailType = "ExampleType"
    }

    val eventsRequest = PutEventsRequest {
        this.entries = listOf(entry)
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putEvents(eventsRequest)
    }
}
```

- Untuk detail API, lihat PutEventsdi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **`PutRule`** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan`PutRule`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat dan picu aturan
- Memulai dengan aturan dan target

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat aturan yang dipicu saat objek ditambahkan ke bucket Amazon Simple Storage Service.

```
    /// <summary>
    /// Create a new event rule that triggers when an Amazon S3 object is created
 in a bucket.
    /// </summary>
    /// <param name="roleArn">The ARN of the role.</param>
    /// <param name="ruleName">The name to give the rule.</param>
    /// <param name="bucketName">The name of the bucket to trigger the event.</
param>
    /// <returns>The ARN of the new rule.</returns>
```

```
    public async Task<string> PutS3UploadRule(string roleArn, string ruleName,
string bucketName)
    {
        string eventPattern = "{" +
                              "\"source\": [\"aws.s3\"]," +
                                  "\"detail-type\": [\"Object Created\"]," +
                                  "\"detail\": {" +
                                      "\"bucket\": {" +
                                          "\"name\": [\"" + bucketName + "\"]"
+
                                      "}" +
                                  "}" +
                              "}";

        var response = await _amazonEventBridge.PutRuleAsync(
            new PutRuleRequest()
            {
                Name = ruleName,
                Description = "Example S3 upload rule for EventBridge",
                RoleArn = roleArn,
                EventPattern = eventPattern
            });

        return response.RuleArn;
    }
```

Buat aturan yang menggunakan pola kustom.

```
    /// <summary>
    /// Update a rule to use a custom defined event pattern.
    /// </summary>
    /// <param name="ruleName">The name of the rule to update.</param>
    /// <returns>The ARN of the updated rule.</returns>
    public async Task<string> UpdateCustomEventPattern(string ruleName)
    {
        string customEventsPattern = "{" +
                                     "\"source\": [\"ExampleSource\"]," +
                                     "\"detail-type\": [\"ExampleType\"]" +
                                     "}";

        var response = await _amazonEventBridge.PutRuleAsync(
            new PutRuleRequest()
```

```
            {
                Name = ruleName,
                Description = "Custom test rule",
                EventPattern = customEventsPattern
            });

        return response.RuleArn;
    }
```

- Untuk detail API, lihat PutRuledi Referensi AWS SDK for .NET API.

C++

SDK for C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Sertakan file-file yang diperlukan.

```
#include <aws/core/Aws.h>
#include <aws/events/EventBridgeClient.h>
#include <aws/events/model/PutRuleRequest.h>
#include <aws/events/model/PutRuleResult.h>
#include <aws/core/utils/Outcome.h>
#include <iostream>
```

Buat aturan.

```
        Aws::CloudWatchEvents::EventBridgeClient cwe;
        Aws::CloudWatchEvents::Model::PutRuleRequest request;
        request.SetName(rule_name);
        request.SetRoleArn(role_arn);
        request.SetScheduleExpression("rate(5 minutes)");
        request.SetState(Aws::CloudWatchEvents::Model::RuleState::ENABLED);
```

```
        auto outcome = cwe.PutRule(request);
        if (!outcome.IsSuccess())
        {
            std::cout << "Failed to create CloudWatch events rule " <<
                rule_name << ": " << outcome.GetError().GetMessage() <<
                std::endl;
        }
        else
        {
            std::cout << "Successfully created CloudWatch events rule " <<
                rule_name << " with resulting Arn " <<
                outcome.GetResult().GetRuleArn() << std::endl;
        }
```

- Untuk detail API, lihat PutRuledi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Untuk membuat aturan CloudWatch Acara

Contoh ini membuat aturan yang dipicu setiap hari pada pukul 9:00 pagi (UTC). Jika Anda menggunakan put-target untuk menambahkan fungsi Lambda sebagai target aturan ini, Anda dapat menjalankan fungsi Lambda setiap hari pada waktu yang ditentukan:

```
aws events put-rule --name "DailyLambdaFunction" --schedule-expression "cron(0 9
 * * ? *)"
```

Contoh ini membuat aturan yang memicu ketika instans EC2 di wilayah mengubah status:

```
aws events put-rule --name "EC2InstanceStateChanges" --event-pattern "{\"source
\":[\"aws.ec2\"],\"detail-type\":[\"EC2 Instance State-change Notification\"]}"
 --role-arn "arn:aws:iam::123456789012:role/MyRoleForThisRule"
```

Contoh ini membuat aturan yang memicu ketika instans EC2 di wilayah tersebut dihentikan atau dihentikan:

```
aws events put-rule --name "EC2InstanceStateChangeStopOrTerminate" --event-
pattern "{\"source\":[\"aws.ec2\"],\"detail-type\":[\"EC2 Instance State-change
```

```
Notification\"],\"detail\":{\"state\":[\"stopped\",\"terminated\"]}}" --role-arn
"arn:aws:iam::123456789012:role/MyRoleForThisRule"
```

- Untuk detail API, lihat PutRuledi Referensi AWS CLI Perintah.

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat aturan terjadwal.

```
   public static void createEBRule(EventBridgeClient eventBrClient, String
ruleName, String cronExpression) {
       try {
           PutRuleRequest ruleRequest = PutRuleRequest.builder()
                   .name(ruleName)
                   .eventBusName("default")
                   .scheduleExpression(cronExpression)
                   .state("ENABLED")
                   .description("A test rule that runs on a schedule created by
the Java API")
                   .build();

           PutRuleResponse ruleResponse = eventBrClient.putRule(ruleRequest);
           System.out.println("The ARN of the new rule is " +
ruleResponse.ruleArn());

       } catch (EventBridgeException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

Buat aturan yang dipicu saat objek ditambahkan ke bucket Amazon Simple Storage Service.

```
   // Create a new event rule that triggers when an Amazon S3 object is created
in
   // a bucket.
   public static void addEventRule(EventBridgeClient eventBrClient, String
roleArn, String bucketName,
           String eventRuleName) {
       String pattern = "{\n" +
               "  \"source\": [\"aws.s3\"],\n" +
               "  \"detail-type\": [\"Object Created\"],\n" +
               "  \"detail\": {\n" +
               "    \"bucket\": {\n" +
               "      \"name\": [\"" + bucketName + "\"]\n" +
               "    }\n" +
               "  }\n" +
               "}";

       try {
           PutRuleRequest ruleRequest = PutRuleRequest.builder()
                   .description("Created by using the AWS SDK for Java v2")
                   .name(eventRuleName)
                   .eventPattern(pattern)
                   .roleArn(roleArn)
                   .build();

           PutRuleResponse ruleResponse = eventBrClient.putRule(ruleRequest);
           System.out.println("The ARN of the new rule is " +
ruleResponse.ruleArn());

       } catch (EventBridgeException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Untuk detail API, lihat PutRuledi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Mengimpor modul SDK dan klien dan memanggil API.

```
import { EventBridgeClient, PutRuleCommand } from "@aws-sdk/client-eventbridge";

export const putRule = async (
  ruleName = "some-rule",
  source = "some-source",
) => {
  const client = new EventBridgeClient({});

  const response = await client.send(
    new PutRuleCommand({
      Name: ruleName,
      EventPattern: JSON.stringify({ source: [source] }),
      State: "ENABLED",
      EventBusName: "default",
    }),
  );

  console.log("PutRule response:");
  console.log(response);
  // PutRule response:
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: 'd7292ced-1544-421b-842f-596326bc7072',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   RuleArn: 'arn:aws:events:us-east-1:xxxxxxxxxxxx:rule/
EventBridgeTestRule-1696280037720'
```

```
  // }
  return response;
};
```

- Untuk detail API, lihat PutRuledi Referensi AWS SDK for JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create CloudWatchEvents service object
var ebevents = new AWS.EventBridge({ apiVersion: "2015-10-07" });

var params = {
  Name: "DEMO_EVENT",
  RoleArn: "IAM_ROLE_ARN",
  ScheduleExpression: "rate(5 minutes)",
  State: "ENABLED",
};

ebevents.putRule(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.RuleArn);
  }
});
```

- Untuk detail API, lihat PutRuledi Referensi AWS SDK for JavaScript API.

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat aturan terjadwal.

```
suspend fun createScRule(ruleName: String?, cronExpression: String?) {
    val ruleRequest = PutRuleRequest {
        name = ruleName
        eventBusName = "default"
        scheduleExpression = cronExpression
        state = RuleState.Enabled
        description = "A test rule that runs on a schedule created by the Kotlin
 API"
    }

    EventBridgeClient { region = "us-west-2" }.use { eventBrClient ->
        val ruleResponse = eventBrClient.putRule(ruleRequest)
        println("The ARN of the new rule is ${ruleResponse.ruleArn}")
    }
}
```

Buat aturan yang dipicu saat objek ditambahkan ke bucket Amazon Simple Storage Service.

```
// Create a new event rule that triggers when an Amazon S3 object is created in a
 bucket.
suspend fun addEventRule(roleArnVal: String?, bucketName: String, eventRuleName:
 String?) {
    val pattern = """{
        "source": ["aws.s3"],
        "detail-type": ["Object Created"],
        "detail": {
        "bucket": {
            "name": ["$bucketName"]
            }
```

```
        }
    }"""

    val ruleRequest = PutRuleRequest {
        description = "Created by using the AWS SDK for Kotlin"
        name = eventRuleName
        eventPattern = pattern
        roleArn = roleArnVal
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val ruleResponse = eventBrClient.putRule(ruleRequest)
        println("The ARN of the new rule is ${ruleResponse.ruleArn}")
    }
}
```

- Untuk detail API, lihat PutRuledi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **`PutTargets`** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan`PutTargets`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- Memulai dengan aturan dan target

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Tambahkan topik Amazon SNS sebagai target aturan.

```
   /// <summary>
   /// Add an Amazon SNS target topic to a rule.
   /// </summary>
   /// <param name="ruleName">The name of the rule to update.</param>
   /// <param name="targetArn">The ARN of the Amazon SNS target.</param>
   /// <param name="eventBusArn">The optional event bus name, uses default if
empty.</param>
   /// <returns>The ID of the target.</returns>
   public async Task<string> AddSnsTargetToRule(string ruleName, string
targetArn, string? eventBusArn = null)
   {
       var targetID = Guid.NewGuid().ToString();

       // Create the list of targets and add a new target.
       var targets = new List<Target>
       {
           new Target()
           {
               Arn = targetArn,
               Id = targetID
           }
       };

       // Add the targets to the rule.
       var response = await _amazonEventBridge.PutTargetsAsync(
           new PutTargetsRequest()
           {
               EventBusName = eventBusArn,
               Rule = ruleName,
               Targets = targets,
           });

       if (response.FailedEntryCount > 0)
       {
           response.FailedEntries.ForEach(e =>
           {
               _logger.LogError(
                   $"Failed to add target {e.TargetId}: {e.ErrorMessage}, code
{e.ErrorCode}");
           });
       }
```

```
        return targetID;
    }
```

Tambahkan transformator input ke target untuk aturan.

```
    /// <summary>
    /// Update an Amazon S3 object created rule with a transform on the target.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <param name="targetArn">The ARN of the target.</param>
    /// <param name="eventBusArn">Optional event bus ARN. If empty, uses the
default event bus.</param>
    /// <returns>The ID of the target.</returns>
    public async Task<string> UpdateS3UploadRuleTargetWithTransform(string
ruleName, string targetArn, string? eventBusArn = null)
    {
        var targetID = Guid.NewGuid().ToString();

        var targets = new List<Target>
        {
            new Target()
            {
                Id = targetID,
                Arn = targetArn,
                InputTransformer = new InputTransformer()
                {
                    InputPathsMap = new Dictionary<string, string>()
                    {
                        {"bucket", "$.detail.bucket.name"},
                        {"time", "$.time"}
                    },
                    InputTemplate = "\"Notification: an object was uploaded to
bucket <bucket> at <time>.\""
                }
            }
        };
        var response = await _amazonEventBridge.PutTargetsAsync(
            new PutTargetsRequest()
            {
                EventBusName = eventBusArn,
                Rule = ruleName,
                Targets = targets,
```

```
            });
        if (response.FailedEntryCount > 0)
        {
            response.FailedEntries.ForEach(e =>
            {
                _logger.LogError(
                    $"Failed to add target {e.TargetId}: {e.ErrorMessage}, code
{e.ErrorCode}");
            });
        }
        return targetID;
    }
```

- Untuk detail API, lihat PutTargetsdi Referensi AWS SDK for .NET API.

C++

SDK for C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Sertakan file-file yang diperlukan.

```
#include <aws/core/Aws.h>
#include <aws/events/EventBridgeClient.h>
#include <aws/events/model/PutTargetsRequest.h>
#include <aws/events/model/PutTargetsResult.h>
#include <aws/core/utils/Outcome.h>
#include <iostream>
```

Tambahkan target.

```
        Aws::CloudWatchEvents::EventBridgeClient cwe;

        Aws::CloudWatchEvents::Model::Target target;
```

```
        target.SetArn(lambda_arn);
        target.SetId(target_id);

        Aws::CloudWatchEvents::Model::PutTargetsRequest request;
        request.SetRule(rule_name);
        request.AddTargets(target);

        auto putTargetsOutcome = cwe.PutTargets(request);
        if (!putTargetsOutcome.IsSuccess())
        {
            std::cout << "Failed to create CloudWatch events target for rule "
                << rule_name << ": " <<
                putTargetsOutcome.GetError().GetMessage() << std::endl;
        }
        else
        {
            std::cout <<
                "Successfully created CloudWatch events target for rule "
                << rule_name << std::endl;
        }
```

- Untuk detail API, lihat PutTargetsdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Untuk menambahkan target untuk aturan CloudWatch Acara

Contoh ini menambahkan fungsi Lambda sebagai target aturan:

```
aws events put-targets --rule DailyLambdaFunction --targets
 "Id"="1","Arn"="arn:aws:lambda:us-east-1:123456789012:function:MyFunctionName"
```

Contoh ini menetapkan aliran Amazon Kinesis sebagai target, sehingga peristiwa yang ditangkap oleh aturan ini diteruskan ke aliran:

```
aws events put-targets --rule EC2InstanceStateChanges --targets
 "Id"="1","Arn"="arn:aws:kinesis:us-east-1:123456789012:stream/
MyStream","RoleArn"="arn:aws:iam::123456789012:role/MyRoleForThisRule"
```

Contoh ini menetapkan dua aliran Amazon Kinesis sebagai target untuk satu aturan:

```
aws events put-targets --rule DailyLambdaFunction --targets
 "Id"="Target1","Arn"="arn:aws:kinesis:us-east-1:379642911888:stream/
MyStream1","RoleArn"="arn:aws:iam::379642911888:role/ MyRoleToAccessLambda"
  "Id"="Target2"," Arn"="arn:aws:kinesis:us-east-1:379642911888:stream/
MyStream2","RoleArn"="arn:aws:iam::379642911888:role/MyRoleToAccessLambda"
```

- Untuk detail API, lihat PutTargetsdi Referensi AWS CLI Perintah.

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Tambahkan topik Amazon SNS sebagai target aturan.

```
    // Add a rule which triggers an SNS target when a file is uploaded to an S3
    // bucket.
    public static void addSnsEventRule(EventBridgeClient eventBrClient, String
ruleName, String topicArn,
            String topicName, String eventRuleName, String bucketName) {
        String targetID = java.util.UUID.randomUUID().toString();
        Target myTarget = Target.builder()
                .id(targetID)
                .arn(topicArn)
                .build();

        List<Target> targets = new ArrayList<>();
        targets.add(myTarget);
        PutTargetsRequest request = PutTargetsRequest.builder()
                .eventBusName(null)
                .targets(targets)
                .rule(ruleName)
                .build();
```

```
        eventBrClient.putTargets(request);
        System.out.println("Added event rule " + eventRuleName + " with Amazon
SNS target " + topicName + " for bucket "
                + bucketName + ".");
    }
```

Tambahkan transformator input ke target untuk aturan.

```
    public static void updateCustomRuleTargetWithTransform(EventBridgeClient
eventBrClient, String topicArn,
            String ruleName) {
        String targetId = java.util.UUID.randomUUID().toString();
        InputTransformer inputTransformer = InputTransformer.builder()
                .inputTemplate("\"Notification: sample event was received.\"")
                .build();

        Target target = Target.builder()
                .id(targetId)
                .arn(topicArn)
                .inputTransformer(inputTransformer)
                .build();

        try {
            PutTargetsRequest targetsRequest = PutTargetsRequest.builder()
                    .rule(ruleName)
                    .targets(target)
                    .eventBusName(null)
                    .build();

            eventBrClient.putTargets(targetsRequest);
        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Untuk detail API, lihat PutTargetsdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Mengimpor modul SDK dan klien dan memanggil API.

```
import {
  EventBridgeClient,
  PutTargetsCommand,
} from "@aws-sdk/client-eventbridge";

export const putTarget = async (
  existingRuleName = "some-rule",
  targetArn = "arn:aws:lambda:us-east-1:000000000000:function:test-func",
  uniqueId = Date.now().toString(),
) => {
  const client = new EventBridgeClient({});
  const response = await client.send(
    new PutTargetsCommand({
      Rule: existingRuleName,
      Targets: [
        {
          Arn: targetArn,
          Id: uniqueId,
        },
      ],
    }),
  );

  console.log("PutTargets response:");
  console.log(response);
  // PutTargets response:
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: 'f5b23b9a-2c17-45c1-ad5c-f926c3692e3d',
  //     extendedRequestId: undefined,
```

```
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   FailedEntries: [],
  //   FailedEntryCount: 0
  // }

  return response;
};
```

- Untuk detail API, lihat PutTargetsdi Referensi AWS SDK for JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create CloudWatchEvents service object
var ebevents = new AWS.EventBridge({ apiVersion: "2015-10-07" });

var params = {
  Rule: "DEMO_EVENT",
  Targets: [
    {
      Arn: "LAMBDA_FUNCTION_ARN",
      Id: "myEventBridgeTarget",
    },
  ],
};

ebevents.putTargets(params, function (err, data) {
  if (err) {
    console.log("Error", err);
```

```
  } else {
    console.log("Success", data);
  }
});
```

- Untuk detail API, lihat PutTargetsdi Referensi AWS SDK for JavaScript API.

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Add a rule that triggers an SNS target when a file is uploaded to an S3
 bucket.
suspend fun addSnsEventRule(ruleName: String?, topicArn: String?, topicName:
 String, eventRuleName: String, bucketName: String) {
    val targetID = UUID.randomUUID().toString()
    val myTarget = Target {
        id = targetID
        arn = topicArn
    }

    val targetsOb = mutableListOf<Target>()
    targetsOb.add(myTarget)

    val request = PutTargetsRequest {
        eventBusName = null
        targets = targetsOb
        rule = ruleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putTargets(request)
        println("Added event rule $eventRuleName with Amazon SNS target
 $topicName for bucket $bucketName.")
    }
```

```
}
```

Tambahkan transformator input ke target untuk aturan.

```
suspend fun updateCustomRuleTargetWithTransform(topicArn: String?, ruleName:
 String?) {
    val targetId = UUID.randomUUID().toString()

    val inputTransformerOb = InputTransformer {
        inputTemplate = "\"Notification: sample event was received.\""
    }

    val target = Target {
        id = targetId
        arn = topicArn
        inputTransformer = inputTransformerOb
    }

    val targetsRequest = PutTargetsRequest {
        rule = ruleName
        targets = listOf(target)
        eventBusName = null
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putTargets(targetsRequest)
    }
}
```

- Untuk detail API, lihat PutTargetsdi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **`RemoveTargets`** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan`RemoveTargets`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Hapus semua target untuk aturan menggunakan nama aturan.

```
    /// <summary>
    /// Delete an event rule by name.
    /// </summary>
    /// <param name="ruleName">The name of the event rule.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> RemoveAllTargetsFromRule(string ruleName)
    {
        var targetIds = new List<string>();
        var request = new ListTargetsByRuleRequest()
        {
            Rule = ruleName
        };
        ListTargetsByRuleResponse targetsResponse;
        do
        {
            targetsResponse = await
_amazonEventBridge.ListTargetsByRuleAsync(request);
            targetIds.AddRange(targetsResponse.Targets.Select(t => t.Id));
            request.NextToken = targetsResponse.NextToken;

        } while (targetsResponse.NextToken is not null);

        var removeResponse = await _amazonEventBridge.RemoveTargetsAsync(
            new RemoveTargetsRequest()
            {
                Rule = ruleName,
                Ids = targetIds
            });

        if (removeResponse.FailedEntryCount > 0)
        {
```

```
            removeResponse.FailedEntries.ForEach(e =>
            {
                _logger.LogError(
                    $"Failed to remove target {e.TargetId}: {e.ErrorMessage},
code {e.ErrorCode}");
            });
        }

        return removeResponse.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat RemoveTargetsdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk menghapus target untuk suatu acara

Contoh ini menghapus aliran Amazon Kinesis bernama MyStream 1 dari target aturan. DailyLambdaFunction Saat DailyLambdaFunction dibuat, aliran ini ditetapkan sebagai target dengan ID Target1:

```
aws events remove-targets --rule "DailyLambdaFunction" --ids "Target1"
```

- Untuk detail API, lihat RemoveTargetsdi Referensi AWS CLI Perintah.

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Hapus semua target untuk aturan dengan menggunakan nama aturan.

```
   public static void deleteTargetsFromRule(EventBridgeClient eventBrClient,
String eventRuleName) {
       // First, get all targets that will be deleted.
       ListTargetsByRuleRequest request = ListTargetsByRuleRequest.builder()
               .rule(eventRuleName)
               .build();

       ListTargetsByRuleResponse response =
eventBrClient.listTargetsByRule(request);
       List<Target> allTargets = response.targets();

       // Get all targets and delete them.
       for (Target myTarget : allTargets) {
           RemoveTargetsRequest removeTargetsRequest =
RemoveTargetsRequest.builder()
                   .rule(eventRuleName)
                   .ids(myTarget.id())
                   .build();

           eventBrClient.removeTargets(removeTargetsRequest);
           System.out.println("Successfully removed the target");
       }
   }
```

- Untuk detail API, lihat RemoveTargetsdi Referensi AWS SDK for Java 2.x API.

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun deleteTargetsFromRule(eventRuleName: String?) {
    // First, get all targets that will be deleted.
    val request = ListTargetsByRuleRequest {
        rule = eventRuleName
```

```
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.listTargetsByRule(request)
        val allTargets = response.targets

        // Get all targets and delete them.
        if (allTargets != null) {
            for (myTarget in allTargets) {
                val removeTargetsRequest = RemoveTargetsRequest {
                    rule = eventRuleName
                    ids = listOf(myTarget.id.toString())
                }
                eventBrClient.removeTargets(removeTargetsRequest)
                println("Successfully removed the target")
            }
        }
    }
}
```

- Untuk detail API, lihat RemoveTargetsdi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Skenario untuk EventBridge menggunakan AWS SDK

Contoh kode berikut menunjukkan cara menerapkan skenario umum EventBridge dengan AWS SDK. Skenario ini menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi di dalamnya EventBridge. Setiap skenario menyertakan tautan ke GitHub, di mana Anda dapat menemukan petunjuk tentang cara mengatur dan menjalankan kode.

Contoh-contoh

- Membuat dan memicu aturan di Amazon EventBridge menggunakan AWS SDK
- Memulai EventBridge aturan dan target menggunakan AWS SDK

Membuat dan memicu aturan di Amazon EventBridge menggunakan AWS SDK

Contoh kode berikut menunjukkan cara membuat dan memicu aturan di Amazon EventBridge.

Ruby

SDK for Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Panggil fungsi dalam urutan yang benar.

```
require "aws-sdk-sns"
require "aws-sdk-iam"
require "aws-sdk-cloudwatchevents"
require "aws-sdk-ec2"
require "aws-sdk-cloudwatch"
require "aws-sdk-cloudwatchlogs"
require "securerandom"
```

Memeriksa apakah topik Amazon Simple Notification Service (Amazon SNS) yang ditentukan ada di antara topik yang disediakan untuk fungsi ini.

```
# Checks whether the specified Amazon SNS
# topic exists among those provided to this function.
# This is a helper function that is called by the topic_exists? function.
#
# @param topics [Array] An array of Aws::SNS::Types::Topic objects.
# @param topic_arn [String] The ARN of the topic to find.
# @return [Boolean] true if the topic ARN was found; otherwise, false.
# @example
#   sns_client = Aws::SNS::Client.new(region: 'us-east-1')
#   response = sns_client.list_topics
#   if topic_found?(
#     response.topics,
```

```
#     'arn:aws:sns:us-east-1:111111111111:aws-doc-sdk-examples-topic'
#   )
#     puts 'Topic found.'
#   end

def topic_found?(topics, topic_arn)
  topics.each do |topic|
    return true if topic.topic_arn == topic_arn
  end
  return false
end
```

Memeriksa apakah topik yang ditentukan ada di antara yang tersedia untuk penelepon di Amazon SNS.

```
# Checks whether the specified topic exists among those available to the
# caller in Amazon SNS.
#
# @param sns_client [Aws::SNS::Client] An initialized Amazon SNS client.
# @param topic_arn [String] The ARN of the topic to find.
# @return [Boolean] true if the topic ARN was found; otherwise, false.
# @example
#   exit 1 unless topic_exists?(
#     Aws::SNS::Client.new(region: 'us-east-1'),
#     'arn:aws:sns:us-east-1:111111111111:aws-doc-sdk-examples-topic'
#   )
def topic_exists?(sns_client, topic_arn)
  puts "Searching for topic with ARN '#{topic_arn}'..."
  response = sns_client.list_topics
  if response.topics.count.positive?
    if topic_found?(response.topics, topic_arn)
      puts "Topic found."
      return true
    end
    while response.next_page? do
      response = response.next_page
      if response.topics.count.positive?
        if topic_found?(response.topics, topic_arn)
          puts "Topic found."
          return true
        end
      end
```

```
    end
  end
  puts "Topic not found."
  return false
rescue StandardError => e
  puts "Topic not found: #{e.message}"
  return false
end
```

Buat topik di Amazon SNS dan kemudian berlangganan alamat email untuk menerima pemberitahuan tentang topik itu.

```
# Creates a topic in Amazon SNS
# and then subscribes an email address to receive notifications to that topic.
#
# @param sns_client [Aws::SNS::Client] An initialized Amazon SNS client.
# @param topic_name [String] The name of the topic to create.
# @param email_address [String] The email address of the recipient to notify.
# @return [String] The ARN of the topic that was created.
# @example
#   puts create_topic(
#     Aws::SNS::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-topic',
#     'mary@example.com'
#   )
def create_topic(sns_client, topic_name, email_address)
  puts "Creating the topic named '#{topic_name}'..."
  topic_response = sns_client.create_topic(name: topic_name)
  puts "Topic created with ARN '#{topic_response.topic_arn}'."
  subscription_response = sns_client.subscribe(
    topic_arn: topic_response.topic_arn,
    protocol: "email",
    endpoint: email_address,
    return_subscription_arn: true
  )
  puts "Subscription created with ARN " \
    "'#{subscription_response.subscription_arn}'. Have the owner of the " \
    "email address '#{email_address}' check their inbox in a few minutes " \
    "and confirm the subscription to start receiving notification emails."
  return topic_response.topic_arn
rescue StandardError => e
  puts "Error creating or subscribing to topic: #{e.message}"
```

```
  return "Error"
end
```

Periksa apakah peran yang ditentukan AWS Identity and Access Management (IAM) ada di antara yang disediakan untuk fungsi ini.

```
# Checks whether the specified AWS Identity and Access Management (IAM)
# role exists among those provided to this function.
# This is a helper function that is called by the role_exists? function.
#
# @param roles [Array] An array of Aws::IAM::Role objects.
# @param role_arn [String] The ARN of the role to find.
# @return [Boolean] true if the role ARN was found; otherwise, false.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-east-1')
#   response = iam_client.list_roles
#   if role_found?(
#     response.roles,
#     'arn:aws:iam::111111111111:role/aws-doc-sdk-examples-ec2-state-change'
#   )
#     puts 'Role found.'
#   end
def role_found?(roles, role_arn)
  roles.each do |role|
    return true if role.arn == role_arn
  end
  return false
end
```

Periksa apakah peran yang ditentukan ada di antara yang tersedia untuk pemanggil di IAM.

```
# Checks whether the specified role exists among those available to the
# caller in AWS Identity and Access Management (IAM).
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param role_arn [String] The ARN of the role to find.
# @return [Boolean] true if the role ARN was found; otherwise, false.
# @example
#   exit 1 unless role_exists?(
#     Aws::IAM::Client.new(region: 'us-east-1'),
#     'arn:aws:iam::111111111111:role/aws-doc-sdk-examples-ec2-state-change'
```

```
#   )
def role_exists?(iam_client, role_arn)
  puts "Searching for role with ARN '#{role_arn}'..."
  response = iam_client.list_roles
  if response.roles.count.positive?
    if role_found?(response.roles, role_arn)
      puts "Role found."
      return true
    end
    while response.next_page? do
      response = response.next_page
      if response.roles.count.positive?
        if role_found?(response.roles, role_arn)
          puts "Role found."
          return true
        end
      end
    end
  end
  puts "Role not found."
  return false
rescue StandardError => e
  puts "Role not found: #{e.message}"
  return false
end
```

Buat peran dalam IAM.

```
# Creates a role in AWS Identity and Access Management (IAM).
# This role is used by a rule in Amazon EventBridge to allow
# that rule to operate within the caller's account.
# This role is designed to be used specifically by this code example.
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param role_name [String] The name of the role to create.
# @return [String] The ARN of the role that was created.
# @example
#   puts create_role(
#     Aws::IAM::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-ec2-state-change'
#   )
def create_role(iam_client, role_name)
```

```
  puts "Creating the role named '#{role_name}'..."
  response = iam_client.create_role(
    assume_role_policy_document: {
      'Version': "2012-10-17",
      'Statement': [
        {
          'Sid': "",
          'Effect': "Allow",
          'Principal': {
            'Service': "events.amazonaws.com"
          },
          'Action': "sts:AssumeRole"
        }
      ]
    }.to_json,
    path: "/",
    role_name: role_name
  )
  puts "Role created with ARN '#{response.role.arn}'."
  puts "Adding access policy to role..."
  iam_client.put_role_policy(
    policy_document: {
      'Version': "2012-10-17",
      'Statement': [
        {
          'Sid': "CloudWatchEventsFullAccess",
          'Effect': "Allow",
          'Resource': "*",
          'Action': "events:*"
        },
        {
          'Sid': "IAMPassRoleForCloudWatchEvents",
          'Effect': "Allow",
          'Resource': "arn:aws:iam::*:role/AWS_Events_Invoke_Targets",
          'Action': "iam:PassRole"
        }
      ]
    }.to_json,
    policy_name: "CloudWatchEventsPolicy",
    role_name: role_name
  )
  puts "Access policy added to role."
  return response.role.arn
rescue StandardError => e
```

```
  puts "Error creating role or adding policy to it: #{e.message}"
  puts "If the role was created, you must add the access policy " \
    "to the role yourself, or delete the role yourself and try again."
  return "Error"
end
```

Memeriksa apakah EventBridge aturan yang ditentukan ada di antara yang disediakan untuk fungsi ini.

```
# Checks whether the specified Amazon EventBridge rule exists among
# those provided to this function.
# This is a helper function that is called by the rule_exists? function.
#
# @param rules [Array] An array of Aws::CloudWatchEvents::Types::Rule objects.
# @param rule_arn [String] The name of the rule to find.
# @return [Boolean] true if the name of the rule was found; otherwise, false.
# @example
#   cloudwatchevents_client = Aws::CloudWatch::Client.new(region: 'us-east-1')
#   response = cloudwatchevents_client.list_rules
#   if rule_found?(response.rules, 'aws-doc-sdk-examples-ec2-state-change')
#     puts 'Rule found.'
#   end
def rule_found?(rules, rule_name)
  rules.each do |rule|
    return true if rule.name == rule_name
  end
  return false
end
```

Memeriksa apakah aturan yang ditentukan ada di antara yang tersedia untuk pemanggil di EventBridge.

```
# Checks whether the specified rule exists among those available to the
# caller in Amazon EventBridge.
#
# @param cloudwatchevents_client [Aws::CloudWatchEvents::Client]
#   An initialized Amazon EventBridge client.
# @param rule_name [String] The name of the rule to find.
# @return [Boolean] true if the rule name was found; otherwise, false.
# @example
```

```
#   exit 1 unless rule_exists?(
#     Aws::CloudWatch::Client.new(region: 'us-east-1')
#     'aws-doc-sdk-examples-ec2-state-change'
#   )
def rule_exists?(cloudwatchevents_client, rule_name)
  puts "Searching for rule with name '#{rule_name}'..."
  response = cloudwatchevents_client.list_rules
  if response.rules.count.positive?
    if rule_found?(response.rules, rule_name)
      puts "Rule found."
      return true
    end
    while response.next_page? do
      response = response.next_page
      if response.rules.count.positive?
        if rule_found?(response.rules, rule_name)
          puts "Rule found."
          return true
        end
      end
    end
  end
  puts "Rule not found."
  return false
rescue StandardError => e
  puts "Rule not found: #{e.message}"
  return false
end
```

Buat aturan di EventBridge.

```
# Creates a rule in Amazon EventBridge.
# This rule is triggered whenever an available instance in
# Amazon EC2 changes to the specified state.
# This rule is designed to be used specifically by this code example.
#
# Prerequisites:
#
# - A role in AWS Identity and Access Management (IAM) that is designed
#   to be used specifically by this code example.
# - A topic in Amazon SNS.
#
```

```
# @param cloudwatchevents_client [Aws::CloudWatchEvents::Client]
#   An initialized Amazon EventBridge client.
# @param rule_name [String] The name of the rule to create.
# @param rule_description [String] Some description for this rule.
# @param instance_state [String] The state that available instances in
#   Amazon EC2 must change to, to
#   trigger this rule.
# @param role_arn [String] The Amazon Resource Name (ARN) of the IAM role.
# @param target_id [String] Some identifying string for the rule's target.
# @param topic_arn [String] The ARN of the Amazon SNS topic.
# @return [Boolean] true if the rule was created; otherwise, false.
# @example
#   exit 1 unless rule_created?(
#     Aws::CloudWatch::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-ec2-state-change',
#     'Triggers when any available EC2 instance starts.',
#     'running',
#     'arn:aws:iam::111111111111:role/aws-doc-sdk-examples-ec2-state-change',
#     'sns-topic',
#     'arn:aws:sns:us-east-1:111111111111:aws-doc-sdk-examples-topic'
#   )
def rule_created?(
  cloudwatchevents_client,
  rule_name,
  rule_description,
  instance_state,
  role_arn,
  target_id,
  topic_arn
)
  puts "Creating rule with name '#{rule_name}'..."
  put_rule_response = cloudwatchevents_client.put_rule(
    name: rule_name,
    description: rule_description,
    event_pattern: {
      'source': [
        "aws.ec2"
      ],
      'detail-type': [
        "EC2 Instance State-change Notification"
      ],
      'detail': {
        'state': [
          instance_state
```

```
        ]
      }
    }.to_json,
    state: "ENABLED",
    role_arn: role_arn
  )
  puts "Rule created with ARN '#{put_rule_response.rule_arn}'."

  put_targets_response = cloudwatchevents_client.put_targets(
    rule: rule_name,
    targets: [
      {
        id: target_id,
        arn: topic_arn
      }
    ]
  )
  if put_targets_response.key?(:failed_entry_count) &&
      put_targets_response.failed_entry_count > 0
    puts "Error(s) adding target to rule:"
    put_targets_response.failed_entries.each do |failure|
      puts failure.error_message
    end
    return false
  else
    return true
  end
rescue StandardError => e
  puts "Error creating rule or adding target to rule: #{e.message}"
  puts "If the rule was created, you must add the target " \
    "to the rule yourself, or delete the rule yourself and try again."
  return false
end
```

Periksa untuk melihat apakah grup log yang ditentukan ada di antara yang tersedia untuk pemanggil di Amazon CloudWatch Logs.

```
# Checks to see whether the specified log group exists among those available
# to the caller in Amazon CloudWatch Logs.
#
# @param cloudwatchlogs_client [Aws::CloudWatchLogs::Client] An initialized
#   Amazon CloudWatch Logs client.
```

```
# @param log_group_name [String] The name of the log group to find.
# @return [Boolean] true if the log group name was found; otherwise, false.
# @example
#   exit 1 unless log_group_exists?(
#     Aws::CloudWatchLogs::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-cloudwatch-log'
#   )
def log_group_exists?(cloudwatchlogs_client, log_group_name)
  puts "Searching for log group with name '#{log_group_name}'..."
  response = cloudwatchlogs_client.describe_log_groups(
    log_group_name_prefix: log_group_name
  )
  if response.log_groups.count.positive?
    response.log_groups.each do |log_group|
      if log_group.log_group_name == log_group_name
        puts "Log group found."
        return true
      end
    end
  end
  puts "Log group not found."
  return false
rescue StandardError => e
  puts "Log group not found: #{e.message}"
  return false
end
```

Buat grup log di CloudWatch Log.

```
# Creates a log group in Amazon CloudWatch Logs.
#
# @param cloudwatchlogs_client [Aws::CloudWatchLogs::Client] An initialized
#   Amazon CloudWatch Logs client.
# @param log_group_name [String] The name of the log group to create.
# @return [Boolean] true if the log group name was created; otherwise, false.
# @example
#   exit 1 unless log_group_created?(
#     Aws::CloudWatchLogs::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-cloudwatch-log'
#   )
def log_group_created?(cloudwatchlogs_client, log_group_name)
  puts "Attempting to create log group with the name '#{log_group_name}'..."
```

```
  cloudwatchlogs_client.create_log_group(log_group_name: log_group_name)
  puts "Log group created."
  return true
rescue StandardError => e
  puts "Error creating log group: #{e.message}"
  return false
end
```

Tulis acara ke aliran log di CloudWatch Log.

```
# Writes an event to a log stream in Amazon CloudWatch Logs.
#
# Prerequisites:
#
# - A log group in Amazon CloudWatch Logs.
# - A log stream within the log group.
#
# @param cloudwatchlogs_client [Aws::CloudWatchLogs::Client] An initialized
#   Amazon CloudWatch Logs client.
# @param log_group_name [String] The name of the log group.
# @param log_stream_name [String] The name of the log stream within
#   the log group.
# @param message [String] The message to write to the log stream.
# @param sequence_token [String] If available, the sequence token from the
#   message that was written immediately before this message. This sequence
#   token is returned by Amazon CloudWatch Logs whenever you programmatically
#   write a message to the log stream.
# @return [String] The sequence token that is returned by
#   Amazon CloudWatch Logs after successfully writing the message to the
#   log stream.
# @example
#   puts log_event(
#     Aws::EC2::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-cloudwatch-log'
#     '2020/11/19/53f985be-199f-408e-9a45-fc242df41fEX',
#     "Instance 'i-033c48ef067af3dEX' restarted.",
#     '495426724868310740095796045676567882148068632824696073EX'
#   )
def log_event(
  cloudwatchlogs_client,
  log_group_name,
  log_stream_name,
```

```
  message,
  sequence_token
)
  puts "Attempting to log '#{message}' to log stream '#{log_stream_name}'..."
  event = {
    log_group_name: log_group_name,
    log_stream_name: log_stream_name,
    log_events: [
      {
        timestamp: (Time.now.utc.to_f.round(3) * 1_000).to_i,
        message: message
      }
    ]
  }
  unless sequence_token.empty?
    event[:sequence_token] = sequence_token
  end

  response = cloudwatchlogs_client.put_log_events(event)
  puts "Message logged."
  return response.next_sequence_token
rescue StandardError => e
  puts "Message not logged: #{e.message}"
end
```

Mulai ulang instance Amazon Elastic Compute Cloud (Amazon EC2) dan tambahkan informasi tentang aktivitas terkait ke aliran log di Log. CloudWatch

```
# Restarts an Amazon EC2 instance
# and adds information about the related activity to a log stream
# in Amazon CloudWatch Logs.
#
# Prerequisites:
#
# - The Amazon EC2 instance to restart.
# - The log group in Amazon CloudWatch Logs to add related activity
#   information to.
#
# @param ec2_client [Aws::EC2::Client] An initialized Amazon EC2 client.
# @param cloudwatchlogs_client [Aws::CloudWatchLogs::Client]
#   An initialized Amazon CloudWatch Logs client.
```

```
# @param instance_id [String] The ID of the instance.
# @param log_group_name [String] The name of the log group.
# @return [Boolean] true if the instance was restarted and the information
#   was written to the log stream; otherwise, false.
# @example
#   exit 1 unless instance_restarted?(
#     Aws::EC2::Client.new(region: 'us-east-1'),
#     Aws::CloudWatchLogs::Client.new(region: 'us-east-1'),
#     'i-033c48ef067af3dEX',
#     'aws-doc-sdk-examples-cloudwatch-log'
#   )
def instance_restarted?(
  ec2_client,
  cloudwatchlogs_client,
  instance_id,
  log_group_name
)
  log_stream_name = "#{Time.now.year}/#{Time.now.month}/#{Time.now.day}/" \
    "#{SecureRandom.uuid}"
  cloudwatchlogs_client.create_log_stream(
    log_group_name: log_group_name,
    log_stream_name: log_stream_name
  )
  sequence_token = ""

  puts "Attempting to stop the instance with the ID '#{instance_id}'. " \
    "This might take a few minutes..."
  ec2_client.stop_instances(instance_ids: [instance_id])
  ec2_client.wait_until(:instance_stopped, instance_ids: [instance_id])
  puts "Instance stopped."
  sequence_token = log_event(
    cloudwatchlogs_client,
    log_group_name,
    log_stream_name,
    "Instance '#{instance_id}' stopped.",
    sequence_token
  )

  puts "Attempting to restart the instance. This might take a few minutes..."
  ec2_client.start_instances(instance_ids: [instance_id])
  ec2_client.wait_until(:instance_running, instance_ids: [instance_id])
  puts "Instance restarted."
  sequence_token = log_event(
    cloudwatchlogs_client,
```

```
    log_group_name,
    log_stream_name,
    "Instance '#{instance_id}' restarted.",
    sequence_token
  )

  return true
rescue StandardError => e
  puts "Error creating log stream or stopping or restarting the instance: " \
    "#{e.message}"
  log_event(
    cloudwatchlogs_client,
    log_group_name,
    log_stream_name,
    "Error stopping or starting instance '#{instance_id}': #{e.message}",
    sequence_token
  )
  return false
end
```

Menampilkan informasi tentang aktivitas untuk aturan di EventBridge.

```
# Displays information about activity for a rule in Amazon EventBridge.
#
# Prerequisites:
#
# - A rule in Amazon EventBridge.
#
# @param cloudwatch_client [Amazon::CloudWatch::Client] An initialized
#   Amazon CloudWatch client.
# @param rule_name [String] The name of the rule.
# @param start_time [Time] The timestamp that determines the first datapoint
#   to return. Can also be expressed as DateTime, Date, Integer, or String.
# @param end_time [Time] The timestamp that determines the last datapoint
#   to return. Can also be expressed as DateTime, Date, Integer, or String.
# @param period [Integer] The interval, in seconds, to check for activity.
# @example
#   display_rule_activity(
#     Aws::CloudWatch::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-ec2-state-change',
#     Time.now - 600, # Start checking from 10 minutes ago.
#     Time.now, # Check up until now.
```

```
#     60 # Check every minute during those 10 minutes.
#   )
def display_rule_activity(
  cloudwatch_client,
  rule_name,
  start_time,
  end_time,
  period
)
  puts "Attempting to display rule activity..."
  response = cloudwatch_client.get_metric_statistics(
    namespace: "AWS/Events",
    metric_name: "Invocations",
    dimensions: [
      {
        name: "RuleName",
        value: rule_name
      }
    ],
    start_time: start_time,
    end_time: end_time,
    period: period,
    statistics: ["Sum"],
    unit: "Count"
  )

  if response.key?(:datapoints) && response.datapoints.count.positive?
    puts "The event rule '#{rule_name}' was triggered:"
    response.datapoints.each do |datapoint|
      puts "  #{datapoint.sum} time(s) at #{datapoint.timestamp}"
    end
  else
    puts "The event rule '#{rule_name}' was not triggered during the " \
      "specified time period."
  end
rescue StandardError => e
  puts "Error getting information about event rule activity: #{e.message}"
end
```

Menampilkan informasi log untuk semua aliran log dalam grup CloudWatch log Log.

```
# Displays log information for all of the log streams in a log group in
```

```
# Amazon CloudWatch Logs.
#
# Prerequisites:
#
# - A log group in Amazon CloudWatch Logs.
#
# @param cloudwatchlogs_client [Amazon::CloudWatchLogs::Client] An initialized
#   Amazon CloudWatch Logs client.
# @param log_group_name [String] The name of the log group.
# @example
#   display_log_data(
#     Amazon::CloudWatchLogs::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-cloudwatch-log'
#   )
def display_log_data(cloudwatchlogs_client, log_group_name)
  puts "Attempting to display log stream data for the log group " \
    "named '#{log_group_name}'..."
  describe_log_streams_response = cloudwatchlogs_client.describe_log_streams(
    log_group_name: log_group_name,
    order_by: "LastEventTime",
    descending: true
  )
  if describe_log_streams_response.key?(:log_streams) &&
      describe_log_streams_response.log_streams.count.positive?
    describe_log_streams_response.log_streams.each do |log_stream|
      get_log_events_response = cloudwatchlogs_client.get_log_events(
        log_group_name: log_group_name,
        log_stream_name: log_stream.log_stream_name
      )
      puts "\nLog messages for '#{log_stream.log_stream_name}':"
      puts "-" * (log_stream.log_stream_name.length + 20)
      if get_log_events_response.key?(:events) &&
          get_log_events_response.events.count.positive?
        get_log_events_response.events.each do |event|
          puts event.message
        end
      else
        puts "No log messages for this log stream."
      end
    end
  end
rescue StandardError => e
  puts "Error getting information about the log streams or their messages: " \
    "#{e.message}"
```

```
end
```

Tampilkan pengingat ke penelepon untuk membersihkan AWS sumber daya terkait secara manual yang tidak lagi mereka butuhkan.

```
# Displays a reminder to the caller to manually clean up any associated
# AWS resources that they no longer need.
#
# @param topic_name [String] The name of the Amazon SNS topic.
# @param role_name [String] The name of the IAM role.
# @param rule_name [String] The name of the Amazon EventBridge rule.
# @param log_group_name [String] The name of the Amazon CloudWatch Logs log
 group.
# @param instance_id [String] The ID of the Amazon EC2 instance.
# @example
#   manual_cleanup_notice(
#     'aws-doc-sdk-examples-topic',
#     'aws-doc-sdk-examples-cloudwatch-events-rule-role',
#     'aws-doc-sdk-examples-ec2-state-change',
#     'aws-doc-sdk-examples-cloudwatch-log',
#     'i-033c48ef067af3dEX'
#   )
def manual_cleanup_notice(
  topic_name, role_name, rule_name, log_group_name, instance_id
)
  puts "-" * 10
  puts "Some of the following AWS resources might still exist in your account."
  puts "If you no longer want to use this code example, then to clean up"
  puts "your AWS account and avoid unexpected costs, you might want to"
  puts "manually delete any of the following resources if they exist:"
  puts "- The Amazon SNS topic named '#{topic_name}'."
  puts "- The IAM role named '#{role_name}'."
  puts "- The Amazon EventBridge rule named '#{rule_name}'."
  puts "- The Amazon CloudWatch Logs log group named '#{log_group_name}'."
  puts "- The Amazon EC2 instance with the ID '#{instance_id}'."
end

# Example usage:
def run_me
  # Properties for the Amazon SNS topic.
  topic_name = "aws-doc-sdk-examples-topic"
```

```
email_address = "mary@example.com"
# Properties for the IAM role.
role_name = "aws-doc-sdk-examples-cloudwatch-events-rule-role"
# Properties for the Amazon EventBridge rule.
rule_name = "aws-doc-sdk-examples-ec2-state-change"
rule_description = "Triggers when any available EC2 instance starts."
instance_state = "running"
target_id = "sns-topic"
# Properties for the Amazon EC2 instance.
instance_id = "i-033c48ef067af3dEX"
# Properties for displaying the event rule's activity.
start_time = Time.now - 600 # Go back over the past 10 minutes
                            # (10 minutes * 60 seconds = 600 seconds).
end_time = Time.now
period = 60 # Look back every 60 seconds over the past 10 minutes.
# Properties for the Amazon CloudWatch Logs log group.
log_group_name = "aws-doc-sdk-examples-cloudwatch-log"
# AWS service clients for this code example.
region = "us-east-1"
sts_client = Aws::STS::Client.new(region: region)
sns_client = Aws::SNS::Client.new(region: region)
iam_client = Aws::IAM::Client.new(region: region)
cloudwatchevents_client = Aws::CloudWatchEvents::Client.new(region: region)
ec2_client = Aws::EC2::Client.new(region: region)
cloudwatch_client = Aws::CloudWatch::Client.new(region: region)
cloudwatchlogs_client = Aws::CloudWatchLogs::Client.new(region: region)

# Get the caller's account ID for use in forming
# Amazon Resource Names (ARNs) that this code relies on later.
account_id = sts_client.get_caller_identity.account

# If the Amazon SNS topic doesn't exist, create it.
topic_arn = "arn:aws:sns:#{region}:#{account_id}:#{topic_name}"
unless topic_exists?(sns_client, topic_arn)
  topic_arn = create_topic(sns_client, topic_name, email_address)
  if topic_arn == "Error"
    puts "Could not create the Amazon SNS topic correctly. Program stopped."
    manual_cleanup_notice(
      topic_name, role_name, rule_name, log_group_name, instance_id
    )
    exit 1
  end
end
```

```
  # If the IAM role doesn't exist, create it.
  role_arn = "arn:aws:iam::#{account_id}:role/#{role_name}"
  unless role_exists?(iam_client, role_arn)
    role_arn = create_role(iam_client, role_name)
    if role_arn == "Error"
      puts "Could not create the IAM role correctly. Program stopped."
      manual_cleanup_notice(
        topic_name, role_name, rule_name, log_group_name, instance_id
      )
    end
  end

  # If the Amazon EventBridge rule doesn't exist, create it.
  unless rule_exists?(cloudwatchevents_client, rule_name)
    unless rule_created?(
      cloudwatchevents_client,
      rule_name,
      rule_description,
      instance_state,
      role_arn,
      target_id,
      topic_arn
    )
      puts "Could not create the Amazon EventBridge rule correctly. " \
        "Program stopped."
      manual_cleanup_notice(
        topic_name, role_name, rule_name, log_group_name, instance_id
      )
    end
  end

  # If the Amazon CloudWatch Logs log group doesn't exist, create it.
  unless log_group_exists?(cloudwatchlogs_client, log_group_name)
    unless log_group_created?(cloudwatchlogs_client, log_group_name)
      puts "Could not create the Amazon CloudWatch Logs log group " \
      "correctly. Program stopped."
      manual_cleanup_notice(
        topic_name, role_name, rule_name, log_group_name, instance_id
      )
    end
  end

  # Restart the Amazon EC2 instance, which triggers the rule.
  unless instance_restarted?(
```

```
    ec2_client,
    cloudwatchlogs_client,
    instance_id,
    log_group_name
  )
    puts "Could not restart the instance to trigger the rule. " \
      "Continuing anyway to show information about the rule and logs..."
  end

  # Display how many times the rule was triggered over the past 10 minutes.
  display_rule_activity(
    cloudwatch_client,
    rule_name,
    start_time,
    end_time,
    period
  )

  # Display related log data in Amazon CloudWatch Logs.
  display_log_data(cloudwatchlogs_client, log_group_name)

  # Reminder the caller to clean up any AWS resources that are used
  # by this code example and are no longer needed.
  manual_cleanup_notice(
    topic_name, role_name, rule_name, log_group_name, instance_id
  )
end

run_me if $PROGRAM_NAME == __FILE__
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for Ruby .
  - PutEvents
  - PutRule

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Memulai EventBridge aturan dan target menggunakan AWS SDK

Contoh-contoh kode berikut menunjukkan cara:

- Buat aturan dan tambahkan target ke dalamnya.
- Aktifkan dan nonaktifkan aturan.
- Daftar dan perbarui aturan dan target.
- Kirim acara, lalu bersihkan sumber daya.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Jalankan skenario interaktif di penggugah/prompt perintah.

```
public class EventBridgeScenario
{
    /*
    Before running this .NET code example, set up your development environment,
 including your credentials.

    This .NET example performs the following tasks with Amazon EventBridge:
    - Create a rule.
    - Add a target to a rule.
    - Enable and disable rules.
    - List rules and targets.
    - Update rules and targets.
    - Send events.
    - Delete the rule.
    */

    private static ILogger logger = null!;
    private static EventBridgeWrapper _eventBridgeWrapper = null!;
    private static IConfiguration _configuration = null!;
```

```
    private static IAmazonIdentityManagementService? _iamClient = null!;
    private static IAmazonSimpleNotificationService? _snsClient = null!;
    private static IAmazonS3 _s3Client = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon EventBridge.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonEventBridge>()
            .AddAWSService<IAmazonIdentityManagementService>()
            .AddAWSService<IAmazonS3>()
            .AddAWSService<IAmazonSimpleNotificationService>()
            .AddTransient<EventBridgeWrapper>()
            )
            .Build();

        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally, load local settings.
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<EventBridgeScenario>();

        ServicesSetup(host);

        string topicArn = "";
        string roleArn = "";

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Welcome to the Amazon EventBridge example scenario.");
        Console.WriteLine(new string('-', 80));

        try
```

```
        {
            roleArn = await CreateRole();

            await CreateBucketWithEventBridgeEvents();

            await AddEventRule(roleArn);

            await ListEventRules();

            topicArn = await CreateSnsTopic();

            var email = await SubscribeToSnsTopic(topicArn);

            await AddSnsTarget(topicArn);

            await ListTargets();

            await ListRulesForTarget(topicArn);

            await UploadS3File(_s3Client);

            await ChangeRuleState(false);

            await GetRuleState();

            await UpdateSnsEventRule(topicArn);

            await ChangeRuleState(true);

            await UploadS3File(_s3Client);

            await UpdateToCustomRule(topicArn);

            await TriggerCustomRule(email);

            await CleanupResources(topicArn);
        }
        catch (Exception ex)
        {
            logger.LogError(ex, "There was a problem executing the scenario.");
            await CleanupResources(topicArn);
        }
        Console.WriteLine(new string('-', 80));
```

```
        Console.WriteLine("The Amazon EventBridge example scenario is
complete.");
        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _eventBridgeWrapper =
host.Services.GetRequiredService<EventBridgeWrapper>();
        _snsClient =
host.Services.GetRequiredService<IAmazonSimpleNotificationService>();
        _s3Client = host.Services.GetRequiredService<IAmazonS3>();
        _iamClient =
host.Services.GetRequiredService<IAmazonIdentityManagementService>();
    }

    /// <summary>
    /// Create a role to be used by EventBridge.
    /// </summary>
    /// <returns>The role Amazon Resource Name (ARN).</returns>
    public static async Task<string> CreateRole()
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Creating a role to use with EventBridge and attaching
managed policy AmazonEventBridgeFullAccess.");
        Console.WriteLine(new string('-', 80));

        var roleName = _configuration["roleName"];

        var assumeRolePolicy = "{" +
                               "\"Version\": \"2012-10-17\"," +
                               "\"Statement\": [{" +
                               "\"Effect\": \"Allow\"," +
                               "\"Principal\": {" +
                               $"\"Service\": \"events.amazonaws.com\"" +
                               "}," +
                               "\"Action\": \"sts:AssumeRole\"" +
                               "}]" +
                               "}";
```

```
        var roleResult = await _iamClient!.CreateRoleAsync(
            new CreateRoleRequest()
            {
                AssumeRolePolicyDocument = assumeRolePolicy,
                Path = "/",
                RoleName = roleName
            });

        await _iamClient.AttachRolePolicyAsync(
            new AttachRolePolicyRequest()
            {
                PolicyArn = "arn:aws:iam::aws:policy/
AmazonEventBridgeFullAccess",
                RoleName = roleName
            });
        // Allow time for the role to be ready.
        Thread.Sleep(10000);
        return roleResult.Role.Arn;
    }

    /// <summary>
    /// Create an Amazon Simple Storage Service (Amazon S3) bucket with
 EventBridge events enabled.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task CreateBucketWithEventBridgeEvents()
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Creating an S3 bucket with EventBridge events
 enabled.");

        var testBucketName = _configuration["testBucketName"];

        var bucketExists = await
 Amazon.S3.Util.AmazonS3Util.DoesS3BucketExistV2Async(_s3Client,
            testBucketName);

        if (!bucketExists)
        {
            await _s3Client.PutBucketAsync(new PutBucketRequest()
            {
                BucketName = testBucketName,
                UseClientRegion = true
            });
```

```
        }

        await _s3Client.PutBucketNotificationAsync(new
PutBucketNotificationRequest()
        {
            BucketName = testBucketName,
            EventBridgeConfiguration = new EventBridgeConfiguration()
        });

        Console.WriteLine($"\tAdded bucket {testBucketName} with EventBridge
events enabled.");

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Create and upload a file to an S3 bucket to trigger an event.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task UploadS3File(IAmazonS3 s3Client)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Uploading a file to the test bucket. This will trigger
a subscription email.");

        var testBucketName = _configuration["testBucketName"];

        var fileName = $"example_upload_{DateTime.UtcNow.Ticks}.txt";

        // Create the file if it does not already exist.
        if (!File.Exists(fileName))
        {
            await using StreamWriter sw = File.CreateText(fileName);
            await sw.WriteLineAsync(
                "This is a sample file for testing uploads.");
        }

        await s3Client.PutObjectAsync(new PutObjectRequest()
        {
            FilePath = fileName,
            BucketName = testBucketName
        });

        Console.WriteLine($"\tPress Enter to continue.");
```

```
        Console.ReadLine();

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Create an Amazon Simple Notification Service (Amazon SNS) topic to use as
an EventBridge target.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task<string> CreateSnsTopic()
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "Creating an Amazon Simple Notification Service (Amazon SNS) topic
for email subscriptions.");

        var topicName = _configuration["topicName"];

        string topicPolicy = "{" +
                             "\"Version\": \"2012-10-17\"," +
                             "\"Statement\": [{" +
                             "\"Sid\": \"EventBridgePublishTopic\"," +
                             "\"Effect\": \"Allow\"," +
                             "\"Principal\": {" +
                             $"\"Service\": \"events.amazonaws.com\"" +
                             "}," +
                             "\"Resource\": \"*\"," +
                             "\"Action\": \"sns:Publish\"" +
                             "}]" +
                             "}";

        var topicAttributes = new Dictionary<string, string>()
        {
            { "Policy", topicPolicy }
        };

        var topicResponse = await _snsClient!.CreateTopicAsync(new
CreateTopicRequest()
        {
            Name = topicName,
            Attributes = topicAttributes

        });
```

```
        Console.WriteLine($"\tAdded topic {topicName} for email subscriptions.");

        Console.WriteLine(new string('-', 80));

        return topicResponse.TopicArn;
    }

    /// <summary>
    /// Subscribe a user email to an SNS topic.
    /// </summary>
    /// <param name="topicArn">The ARN of the SNS topic.</param>
    /// <returns>The user's email.</returns>
    private static async Task<string> SubscribeToSnsTopic(string topicArn)
    {
        Console.WriteLine(new string('-', 80));


        string email = "";
        while (string.IsNullOrEmpty(email))
        {
            Console.WriteLine("Enter your email to subscribe to the Amazon SNS
topic:");
            email = Console.ReadLine()!;
        }

        var subscriptions = new List<string>();
        var paginatedSubscriptions =
_snsClient!.Paginators.ListSubscriptionsByTopic(
            new ListSubscriptionsByTopicRequest()
            {
                TopicArn = topicArn
            });

        // Get the entire list using the paginator.
        await foreach (var subscription in paginatedSubscriptions.Subscriptions)
        {
            subscriptions.Add(subscription.Endpoint);
        }

        if (subscriptions.Contains(email))
        {
            Console.WriteLine($"\tYour email is already subscribed.");
            Console.WriteLine(new string('-', 80));
```

```
            return email;
        }

        await _snsClient.SubscribeAsync(new SubscribeRequest()
        {
            TopicArn = topicArn,
            Protocol = "email",
            Endpoint = email
        });

        Console.WriteLine($"Use the link in the email you received to confirm
your subscription, then press Enter to continue.");

        Console.ReadLine();

        Console.WriteLine(new string('-', 80));
        return email;
    }

    /// <summary>
    /// Add a rule which triggers when a file is uploaded to an S3 bucket.
    /// </summary>
    /// <param name="roleArn">The ARN of the role used by EventBridge.</param>
    /// <returns>Async task.</returns>
    private static async Task AddEventRule(string roleArn)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Creating an EventBridge event that sends an email when
an Amazon S3 object is created.");

        var eventRuleName = _configuration["eventRuleName"];
        var testBucketName = _configuration["testBucketName"];

        await _eventBridgeWrapper.PutS3UploadRule(roleArn, eventRuleName,
testBucketName);
        Console.WriteLine($"\tAdded event rule {eventRuleName} for bucket
{testBucketName}.");

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Add an SNS target to the rule.
    /// </summary>
```

```
    /// <param name="topicArn">The ARN of the SNS topic.</param>
    /// <returns>Async task.</returns>
    private static async Task AddSnsTarget(string topicArn)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Adding a target to the rule to that sends an email
when the rule is triggered.");

        var eventRuleName = _configuration["eventRuleName"];
        var testBucketName = _configuration["testBucketName"];
        var topicName = _configuration["topicName"];
        await _eventBridgeWrapper.AddSnsTargetToRule(eventRuleName, topicArn);
        Console.WriteLine($"\tAdded event rule {eventRuleName} with Amazon SNS
target {topicName} for bucket {testBucketName}.");

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// List the event rules on the default event bus.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task ListEventRules()
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Current event rules:");

        var rules = await _eventBridgeWrapper.ListAllRulesForEventBus();
        rules.ForEach(r => Console.WriteLine($"\tRule: {r.Name} Description:
{r.Description} State: {r.State}"));

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Update the event target to use a transform.
    /// </summary>
    /// <param name="topicArn">The SNS topic ARN target to update.</param>
    /// <returns>Async task.</returns>
    private static async Task UpdateSnsEventRule(string topicArn)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Let's update the event target with a transform.");
```

```
        var eventRuleName = _configuration["eventRuleName"];
        var testBucketName = _configuration["testBucketName"];

        await
_eventBridgeWrapper.UpdateS3UploadRuleTargetWithTransform(eventRuleName,
topicArn);
        Console.WriteLine($"\tUpdated event rule {eventRuleName} with Amazon SNS
target {topicArn} for bucket {testBucketName}.");

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Update the rule to use a custom event pattern.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task UpdateToCustomRule(string topicArn)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Updating the event pattern to be triggered by a custom
event instead.");

        var eventRuleName = _configuration["eventRuleName"];

        await _eventBridgeWrapper.UpdateCustomEventPattern(eventRuleName);

        Console.WriteLine($"\tUpdated event rule {eventRuleName} to custom
pattern.");
        await
_eventBridgeWrapper.UpdateCustomRuleTargetWithTransform(eventRuleName,
            topicArn);

        Console.WriteLine($"\tUpdated event target {topicArn}.");

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Send rule events for a custom rule using the user's email address.
    /// </summary>
    /// <param name="email">The email address to include.</param>
    /// <returns>Async task.</returns>
    private static async Task TriggerCustomRule(string email)
    {
```

```
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Sending an event to trigger the rule. This will
trigger a subscription email.");

        await _eventBridgeWrapper.PutCustomEmailEvent(email);

        Console.WriteLine($"\tEvents have been sent. Press Enter to continue.");
        Console.ReadLine();

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// List all of the targets for a rule.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task ListTargets()
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("List all of the targets for a particular rule.");

        var eventRuleName = _configuration["eventRuleName"];
        var targets = await
_eventBridgeWrapper.ListAllTargetsOnRule(eventRuleName);
        targets.ForEach(t => Console.WriteLine($"\tTarget: {t.Arn} Id: {t.Id}
Input: {t.Input}"));

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// List all of the rules for a particular target.
    /// </summary>
    /// <param name="topicArn">The ARN of the SNS topic.</param>
    /// <returns>Async task.</returns>
    private static async Task ListRulesForTarget(string topicArn)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("List all of the rules for a particular target.");

        var rules = await _eventBridgeWrapper.ListAllRuleNamesByTarget(topicArn);
        rules.ForEach(r => Console.WriteLine($"\tRule: {r}"));

        Console.WriteLine(new string('-', 80));
```

```
    }

    /// <summary>
    /// Enable or disable a particular rule.
    /// </summary>
    /// <param name="isEnabled">True to enable the rule, otherwise false.</param>
    /// <returns>Async task.</returns>
    private static async Task ChangeRuleState(bool isEnabled)
    {
        Console.WriteLine(new string('-', 80));
        var eventRuleName = _configuration["eventRuleName"];

        if (!isEnabled)
        {
            Console.WriteLine($"Disabling the rule: {eventRuleName}");
            await _eventBridgeWrapper.DisableRuleByName(eventRuleName);
        }
        else
        {
            Console.WriteLine($"Enabling the rule: {eventRuleName}");
            await _eventBridgeWrapper.EnableRuleByName(eventRuleName);
        }

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Get the current state of the rule.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task GetRuleState()
    {
        Console.WriteLine(new string('-', 80));
        var eventRuleName = _configuration["eventRuleName"];

        var state = await
_eventBridgeWrapper.GetRuleStateByRuleName(eventRuleName);
        Console.WriteLine($"Rule {eventRuleName} is in current state {state}.");

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Clean up the resources from the scenario.
```

```
    /// </summary>
    /// <param name="topicArn">The ARN of the SNS topic to clean up.</param>
    /// <returns>Async task.</returns>
    private static async Task CleanupResources(string topicArn)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine($"Clean up resources.");

        var eventRuleName = _configuration["eventRuleName"];
        if (GetYesNoResponse($"\tDelete all targets and event rule
{eventRuleName}? (y/n)"))
        {
            Console.WriteLine($"\tRemoving all targets from the event rule.");
            await _eventBridgeWrapper.RemoveAllTargetsFromRule(eventRuleName);

            Console.WriteLine($"\tDeleting event rule.");
            await _eventBridgeWrapper.DeleteRuleByName(eventRuleName);
        }

        var topicName = _configuration["topicName"];
        if (GetYesNoResponse($"\tDelete Amazon SNS subscription topic
{topicName}? (y/n)"))
        {
            Console.WriteLine($"\tDeleting topic.");
            await _snsClient!.DeleteTopicAsync(new DeleteTopicRequest()
            {
                TopicArn = topicArn
            });
        }

        var bucketName = _configuration["testBucketName"];
        if (GetYesNoResponse($"\tDelete Amazon S3 bucket {bucketName}? (y/n)"))
        {
            Console.WriteLine($"\tDeleting bucket.");
            // Delete all objects in the bucket.
            var deleteList = await _s3Client.ListObjectsV2Async(new
ListObjectsV2Request()
            {
                BucketName = bucketName
            });
            await _s3Client.DeleteObjectsAsync(new DeleteObjectsRequest()
            {
                BucketName = bucketName,
                Objects = deleteList.S3Objects
```

```
                    .Select(o => new KeyVersion { Key = o.Key }).ToList()
            });
            // Now delete the bucket.
            await _s3Client.DeleteBucketAsync(new DeleteBucketRequest()
            {
                BucketName = bucketName
            });
        }

        var roleName = _configuration["roleName"];
        if (GetYesNoResponse($"\tDelete role {roleName}? (y/n)"))
        {
            Console.WriteLine($"\tDetaching policy and deleting role.");

            await _iamClient!.DetachRolePolicyAsync(new DetachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = "arn:aws:iam::aws:policy/
AmazonEventBridgeFullAccess",
            });

            await _iamClient!.DeleteRoleAsync(new DeleteRoleRequest()
            {
                RoleName = roleName
            });
        }

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Helper method to get a yes or no response from the user.
    /// </summary>
    /// <param name="question">The question string to print on the console.</
param>
    /// <returns>True if the user responds with a yes.</returns>
    private static bool GetYesNoResponse(string question)
    {
        Console.WriteLine(question);
        var ynResponse = Console.ReadLine();
        var response = ynResponse != null &&
                       ynResponse.Equals("y",
                           StringComparison.InvariantCultureIgnoreCase);
        return response;
```

```
    }
}
```

Buat kelas yang membungkus EventBridge operasi.

```
/// <summary>
/// Wrapper for Amazon EventBridge operations.
/// </summary>
public class EventBridgeWrapper
{
    private readonly IAmazonEventBridge _amazonEventBridge;
    private readonly ILogger<EventBridgeWrapper> _logger;

    /// <summary>
    /// Constructor for the EventBridge wrapper.
    /// </summary>
    /// <param name="amazonEventBridge">The injected EventBridge client.</param>
    /// <param name="logger">The injected logger for the wrapper.</param>
    public EventBridgeWrapper(IAmazonEventBridge amazonEventBridge,
 ILogger<EventBridgeWrapper> logger)

    {
        _amazonEventBridge = amazonEventBridge;
        _logger = logger;
    }

    /// <summary>
    /// Get the state for a rule by the rule name.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <param name="eventBusName">The optional name of the event bus. If empty,
 uses the default event bus.</param>
    /// <returns>The state of the rule.</returns>
    public async Task<RuleState> GetRuleStateByRuleName(string ruleName, string?
 eventBusName = null)
    {
        var ruleResponse = await _amazonEventBridge.DescribeRuleAsync(
            new DescribeRuleRequest()
            {
                Name = ruleName,
                EventBusName = eventBusName
```

```
            });
        return ruleResponse.State;
    }

    /// <summary>
    /// Enable a particular rule on an event bus.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> EnableRuleByName(string ruleName)
    {
        var ruleResponse = await _amazonEventBridge.EnableRuleAsync(
            new EnableRuleRequest()
            {
                Name = ruleName
            });
        return ruleResponse.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Disable a particular rule on an event bus.
    /// </summary
    /// <param name="ruleName">The name of the rule.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DisableRuleByName(string ruleName)
    {
        var ruleResponse = await _amazonEventBridge.DisableRuleAsync(
            new DisableRuleRequest()
            {
                Name = ruleName
            });
        return ruleResponse.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the rules on an event bus.
    /// </summary>
    /// <param name="eventBusArn">The optional ARN of the event bus. If empty,
uses the default event bus.</param>
    /// <returns>The list of rules.</returns>
    public async Task<List<Rule>> ListAllRulesForEventBus(string? eventBusArn =
null)
    {
        var results = new List<Rule>();
```

```
        var request = new ListRulesRequest()
        {
            EventBusName = eventBusArn
        };
        // Get all of the pages of rules.
        ListRulesResponse response;
        do
        {
            response = await _amazonEventBridge.ListRulesAsync(request);
            results.AddRange(response.Rules);
            request.NextToken = response.NextToken;

        } while (response.NextToken is not null);

        return results;
    }

    /// <summary>
    /// List all of the targets matching a rule by name.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <returns>The list of targets.</returns>
    public async Task<List<Target>> ListAllTargetsOnRule(string ruleName)
    {
        var results = new List<Target>();
        var request = new ListTargetsByRuleRequest()
        {
            Rule = ruleName
        };
        ListTargetsByRuleResponse response;
        do
        {
            response = await _amazonEventBridge.ListTargetsByRuleAsync(request);
            results.AddRange(response.Targets);
            request.NextToken = response.NextToken;

        } while (response.NextToken is not null);

        return results;
    }

    /// <summary>
    /// List names of all rules matching a target.
    /// </summary>
```

```
    /// <param name="targetArn">The ARN of the target.</param>
    /// <returns>The list of rule names.</returns>
    public async Task<List<string>> ListAllRuleNamesByTarget(string targetArn)
    {
        var results = new List<string>();
        var request = new ListRuleNamesByTargetRequest()
        {
            TargetArn = targetArn
        };
        ListRuleNamesByTargetResponse response;
        do
        {
            response = await
 _amazonEventBridge.ListRuleNamesByTargetAsync(request);
            results.AddRange(response.RuleNames);
            request.NextToken = response.NextToken;

        } while (response.NextToken is not null);

        return results;
    }

    /// <summary>
    /// Create a new event rule that triggers when an Amazon S3 object is created
 in a bucket.
    /// </summary>
    /// <param name="roleArn">The ARN of the role.</param>
    /// <param name="ruleName">The name to give the rule.</param>
    /// <param name="bucketName">The name of the bucket to trigger the event.</
param>
    /// <returns>The ARN of the new rule.</returns>
    public async Task<string> PutS3UploadRule(string roleArn, string ruleName,
 string bucketName)
    {
        string eventPattern = "{" +
                              "\"source\": [\"aws.s3\"]," +
                                  "\"detail-type\": [\"Object Created\"]," +
                                  "\"detail\": {" +
                                      "\"bucket\": {" +
                                          "\"name\": [\"" + bucketName + "\"]"
 +
                                      "}" +
                                  "}" +
                              "}";
```

```
        var response = await _amazonEventBridge.PutRuleAsync(
            new PutRuleRequest()
            {
                Name = ruleName,
                Description = "Example S3 upload rule for EventBridge",
                RoleArn = roleArn,
                EventPattern = eventPattern
            });

        return response.RuleArn;
    }

    /// <summary>
    /// Update an Amazon S3 object created rule with a transform on the target.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <param name="targetArn">The ARN of the target.</param>
    /// <param name="eventBusArn">Optional event bus ARN. If empty, uses the
default event bus.</param>
    /// <returns>The ID of the target.</returns>
    public async Task<string> UpdateS3UploadRuleTargetWithTransform(string
ruleName, string targetArn, string? eventBusArn = null)
    {
        var targetID = Guid.NewGuid().ToString();

        var targets = new List<Target>
        {
            new Target()
            {
                Id = targetID,
                Arn = targetArn,
                InputTransformer = new InputTransformer()
                {
                    InputPathsMap = new Dictionary<string, string>()
                    {
                        {"bucket", "$.detail.bucket.name"},
                        {"time", "$.time"}
                    },
                    InputTemplate = "\"Notification: an object was uploaded to
bucket <bucket> at <time>.\""
                }
            }
        };
```

```
        var response = await _amazonEventBridge.PutTargetsAsync(
            new PutTargetsRequest()
            {
                EventBusName = eventBusArn,
                Rule = ruleName,
                Targets = targets,
            });
        if (response.FailedEntryCount > 0)
        {
            response.FailedEntries.ForEach(e =>
            {
                _logger.LogError(
                    $"Failed to add target {e.TargetId}: {e.ErrorMessage}, code
 {e.ErrorCode}");
            });
        }
        return targetID;
    }

    /// <summary>
    /// Update a custom rule with a transform on the target.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <param name="targetArn">The ARN of the target.</param>
    /// <param name="eventBusArn">Optional event bus ARN. If empty, uses the
 default event bus.</param>
    /// <returns>The ID of the target.</returns>
    public async Task<string> UpdateCustomRuleTargetWithTransform(string
 ruleName, string targetArn, string? eventBusArn = null)
    {
        var targetID = Guid.NewGuid().ToString();

        var targets = new List<Target>
        {
            new Target()
            {
                Id = targetID,
                Arn = targetArn,
                InputTransformer = new InputTransformer()
                {
                    InputTemplate = "\"Notification: sample event was received.
\""
                }
            }
```

```
        };
        var response = await _amazonEventBridge.PutTargetsAsync(
            new PutTargetsRequest()
            {
                EventBusName = eventBusArn,
                Rule = ruleName,
                Targets = targets,
            });
        if (response.FailedEntryCount > 0)
        {
            response.FailedEntries.ForEach(e =>
            {
                _logger.LogError(
                    $"Failed to add target {e.TargetId}: {e.ErrorMessage}, code
{e.ErrorCode}");
            });
        }
        return targetID;
    }

    /// <summary>
    /// Add an event to the event bus that includes an email, message, and time.
    /// </summary>
    /// <param name="email">The email to use in the event detail of the custom
event.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> PutCustomEmailEvent(string email)
    {
        var eventDetail = new
        {
            UserEmail = email,
            Message = "This event was generated by example code.",
            UtcTime = DateTime.UtcNow.ToString("g")
        };
        var response = await _amazonEventBridge.PutEventsAsync(
            new PutEventsRequest()
            {
                Entries = new List<PutEventsRequestEntry>()
                {
                    new PutEventsRequestEntry()
                    {
                        Source = "ExampleSource",
                        Detail = JsonSerializer.Serialize(eventDetail),
                        DetailType = "ExampleType"
```

```
                    }
                }
            });

        return response.FailedEntryCount == 0;
    }

    /// <summary>
    /// Update a rule to use a custom defined event pattern.
    /// </summary>
    /// <param name="ruleName">The name of the rule to update.</param>
    /// <returns>The ARN of the updated rule.</returns>
    public async Task<string> UpdateCustomEventPattern(string ruleName)
    {
        string customEventsPattern = "{" +
                                     "\"source\": [\"ExampleSource\"]," +
                                     "\"detail-type\": [\"ExampleType\"]" +
                                     "}";

        var response = await _amazonEventBridge.PutRuleAsync(
            new PutRuleRequest()
            {
                Name = ruleName,
                Description = "Custom test rule",
                EventPattern = customEventsPattern
            });

        return response.RuleArn;
    }

    /// <summary>
    /// Add an Amazon SNS target topic to a rule.
    /// </summary>
    /// <param name="ruleName">The name of the rule to update.</param>
    /// <param name="targetArn">The ARN of the Amazon SNS target.</param>
    /// <param name="eventBusArn">The optional event bus name, uses default if
empty.</param>
    /// <returns>The ID of the target.</returns>
    public async Task<string> AddSnsTargetToRule(string ruleName, string
targetArn, string? eventBusArn = null)
    {
        var targetID = Guid.NewGuid().ToString();

        // Create the list of targets and add a new target.
```

```
        var targets = new List<Target>
        {
            new Target()
            {
                Arn = targetArn,
                Id = targetID
            }
        };

        // Add the targets to the rule.
        var response = await _amazonEventBridge.PutTargetsAsync(
            new PutTargetsRequest()
            {
                EventBusName = eventBusArn,
                Rule = ruleName,
                Targets = targets,
            });

        if (response.FailedEntryCount > 0)
        {
            response.FailedEntries.ForEach(e =>
            {
                _logger.LogError(
                    $"Failed to add target {e.TargetId}: {e.ErrorMessage}, code
{e.ErrorCode}");
            });
        }

        return targetID;
    }

    /// <summary>
    /// Delete an event rule by name.
    /// </summary>
    /// <param name="ruleName">The name of the event rule.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> RemoveAllTargetsFromRule(string ruleName)
    {
        var targetIds = new List<string>();
        var request = new ListTargetsByRuleRequest()
        {
            Rule = ruleName
        };
        ListTargetsByRuleResponse targetsResponse;
```

```
        do
        {
            targetsResponse = await
_amazonEventBridge.ListTargetsByRuleAsync(request);
            targetIds.AddRange(targetsResponse.Targets.Select(t => t.Id));
            request.NextToken = targetsResponse.NextToken;

        } while (targetsResponse.NextToken is not null);

        var removeResponse = await _amazonEventBridge.RemoveTargetsAsync(
            new RemoveTargetsRequest()
            {
                Rule = ruleName,
                Ids = targetIds
            });

        if (removeResponse.FailedEntryCount > 0)
        {
            removeResponse.FailedEntries.ForEach(e =>
            {
                _logger.LogError(
                    $"Failed to remove target {e.TargetId}: {e.ErrorMessage},
code {e.ErrorCode}");
            });
        }

        return removeResponse.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Delete an event rule by name.
    /// </summary>
    /// <param name="ruleName">The name of the event rule.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteRuleByName(string ruleName)
    {
        var response = await _amazonEventBridge.DeleteRuleAsync(
            new DeleteRuleRequest()
            {
                Name = ruleName
            });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for .NET .
  - DeleteRule
  - DescribeRule
  - DisableRule
  - EnableRule
  - ListRuleNamesByTarget
  - ListRules
  - ListTargetsByRule
  - PutEvents
  - PutRule
  - PutTargets

Java

SDK for Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * This Java code example performs the following tasks:
 *
 * This Java V2 example performs the following tasks with Amazon EventBridge:
```

```
 *
 * 1. Creates an AWS Identity and Access Management (IAM) role to use with
 * Amazon EventBridge.
 * 2. Amazon Simple Storage Service (Amazon S3) bucket with EventBridge events
 * enabled.
 * 3. Creates a rule that triggers when an object is uploaded to Amazon S3.
 * 4. Lists rules on the event bus.
 * 5. Creates a new Amazon Simple Notification Service (Amazon SNS) topic and
 * lets the user subscribe to it.
 * 6. Adds a target to the rule that sends an email to the specified topic.
 * 7. Creates an EventBridge event that sends an email when an Amazon S3 object
 * is created.
 * 8. Lists Targets.
 * 9. Lists the rules for the same target.
 * 10. Triggers the rule by uploading a file to the Amazon S3 bucket.
 * 11. Disables a specific rule.
 * 12. Checks and print the state of the rule.
 * 13. Adds a transform to the rule to change the text of the email.
 * 14. Enables a specific rule.
 * 15. Triggers the updated rule by uploading a file to the Amazon S3 bucket.
 * 16. Updates the rule to be a custom rule pattern.
 * 17. Sending an event to trigger the rule.
 * 18. Cleans up resources.
 *
 */
public class EventbridgeMVP {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws InterruptedException,
 IOException {
        final String usage = """

                Usage:
                    <roleName> <bucketName> <topicName> <eventRuleName>

                Where:
                    roleName - The name of the role to create.
                    bucketName - The Amazon Simple Storage Service (Amazon S3)
 bucket name to create.
                    topicName - The name of the Amazon Simple Notification
 Service (Amazon SNS) topic to create.
                    eventRuleName - The Amazon EventBridge rule name to create.
                """;
```

```
        if (args.length != 5) {
            System.out.println(usage);
            System.exit(1);
        }

        String polJSON = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                "\"Service\": \"events.amazonaws.com\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        Scanner sc = new Scanner(System.in);
        String roleName = args[0];
        String bucketName = args[1];
        String topicName = args[2];
        String eventRuleName = args[3];

        Region region = Region.US_EAST_1;
        EventBridgeClient eventBrClient = EventBridgeClient.builder()
                .region(region)
                .build();

        S3Client s3Client = S3Client.builder()
                .region(region)
                .build();

        Region regionGl = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(regionGl)
                .build();

        SnsClient snsClient = SnsClient.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon EventBridge example
scenario.");
```

```
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out
                .println("1. Create an AWS Identity and Access Management (IAM)
role to use with Amazon EventBridge.");
        String roleArn = createIAMRole(iam, roleName, polJSON);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("2. Create an S3 bucket with EventBridge events
enabled.");
        if (checkBucket(s3Client, bucketName)) {
            System.out.println("Bucket " + bucketName + " already exists. Ending
this scenario.");
            System.exit(1);
        }

        createBucket(s3Client, bucketName);
        Thread.sleep(3000);
        setBucketNotification(s3Client, bucketName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Create a rule that triggers when an object is
uploaded to Amazon S3.");
        Thread.sleep(10000);
        addEventRule(eventBrClient, roleArn, bucketName, eventRuleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. List rules on the event bus.");
        listRules(eventBrClient);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("5. Create a new SNS topic for testing and let the
user subscribe to the topic.");
        String topicArn = createSnsTopic(snsClient, topicName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6. Add a target to the rule that sends an email to
the specified topic.");
```

```
        System.out.println("Enter your email to subscribe to the Amazon SNS
topic:");
        String email = sc.nextLine();
        subEmail(snsClient, topicArn, email);
        System.out.println(
                "Use the link in the email you received to confirm your
subscription. Then, press Enter to continue.");
        sc.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Create an EventBridge event that sends an email
when an Amazon S3 object is created.");
        addSnsEventRule(eventBrClient, eventRuleName, topicArn, topicName,
eventRuleName, bucketName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 8. List Targets.");
        listTargets(eventBrClient, eventRuleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 9. List the rules for the same target.");
        listTargetRules(eventBrClient, topicArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 10. Trigger the rule by uploading a file to the S3
bucket.");
        System.out.println("Press Enter to continue.");
        sc.nextLine();
        uploadTextFiletoS3(s3Client, bucketName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("11. Disable a specific rule.");
        changeRuleState(eventBrClient, eventRuleName, false);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("12. Check and print the state of the rule.");
        checkRule(eventBrClient, eventRuleName);
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("13. Add a transform to the rule to change the text of
the email.");
        updateSnsEventRule(eventBrClient, topicArn, eventRuleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("14. Enable a specific rule.");
        changeRuleState(eventBrClient, eventRuleName, true);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 15. Trigger the updated rule by uploading a file to
the S3 bucket.");
        System.out.println("Press Enter to continue.");
        sc.nextLine();
        uploadTextFiletoS3(s3Client, bucketName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 16. Update the rule to be a custom rule pattern.");
        updateToCustomRule(eventBrClient, eventRuleName);
        System.out.println("Updated event rule " + eventRuleName + " to use a
custom pattern.");
        updateCustomRuleTargetWithTransform(eventBrClient, topicArn,
eventRuleName);
        System.out.println("Updated event target " + topicArn + ".");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("17. Sending an event to trigger the rule. This will
trigger a subscription email.");
        triggerCustomRule(eventBrClient, email);
        System.out.println("Events have been sent. Press Enter to continue.");
        sc.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("18. Clean up resources.");
        System.out.println("Do you want to clean up resources (y/n)");
        String ans = sc.nextLine();
        if (ans.compareTo("y") == 0) {
```

```
            cleanupResources(eventBrClient, snsClient, s3Client, iam, topicArn,
eventRuleName, bucketName, roleName);
        } else {
            System.out.println("The resources will not be cleaned up. ");
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The Amazon EventBridge example scenario has
successfully completed.");
        System.out.println(DASHES);
    }

    public static void cleanupResources(EventBridgeClient eventBrClient,
SnsClient snsClient, S3Client s3Client,
            IamClient iam, String topicArn, String eventRuleName, String
bucketName, String roleName) {
        System.out.println("Removing all targets from the event rule.");
        deleteTargetsFromRule(eventBrClient, eventRuleName);
        deleteRuleByName(eventBrClient, eventRuleName);
        deleteSNSTopic(snsClient, topicArn);
        deleteS3Bucket(s3Client, bucketName);
        deleteRole(iam, roleName);
    }

    public static void deleteRole(IamClient iam, String roleName) {
        String policyArn = "arn:aws:iam::aws:policy/AmazonEventBridgeFullAccess";
        DetachRolePolicyRequest policyRequest = DetachRolePolicyRequest.builder()
                .policyArn(policyArn)
                .roleName(roleName)
                .build();

        iam.detachRolePolicy(policyRequest);
        System.out.println("Successfully detached policy " + policyArn + " from
role " + roleName);

        // Delete the role.
        DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                .roleName(roleName)
                .build();

        iam.deleteRole(roleRequest);
        System.out.println("*** Successfully deleted " + roleName);
    }
```

```
    public static void deleteS3Bucket(S3Client s3Client, String bucketName) {
        // Remove all the objects from the S3 bucket.
        ListObjectsRequest listObjects = ListObjectsRequest.builder()
                .bucket(bucketName)
                .build();

        ListObjectsResponse res = s3Client.listObjects(listObjects);
        List<S3Object> objects = res.contents();
        ArrayList<ObjectIdentifier> toDelete = new ArrayList<>();

        for (S3Object myValue : objects) {
            toDelete.add(ObjectIdentifier.builder()
                    .key(myValue.key())
                    .build());
        }

        DeleteObjectsRequest dor = DeleteObjectsRequest.builder()
                .bucket(bucketName)
                .delete(Delete.builder()
                        .objects(toDelete).build())
                .build();

        s3Client.deleteObjects(dor);

        // Delete the S3 bucket.
        DeleteBucketRequest deleteBucketRequest = DeleteBucketRequest.builder()
                .bucket(bucketName)
                .build();

        s3Client.deleteBucket(deleteBucketRequest);
        System.out.println("You have deleted the bucket and the objects");
    }

    // Delete the SNS topic.
    public static void deleteSNSTopic(SnsClient snsClient, String topicArn) {
        try {
            DeleteTopicRequest request = DeleteTopicRequest.builder()
                    .topicArn(topicArn)
                    .build();

            DeleteTopicResponse result = snsClient.deleteTopic(request);
            System.out.println("\n\nStatus was " +
result.sdkHttpResponse().statusCode());
```

```
        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteRuleByName(EventBridgeClient eventBrClient, String
ruleName) {
        DeleteRuleRequest ruleRequest = DeleteRuleRequest.builder()
                .name(ruleName)
                .build();

        eventBrClient.deleteRule(ruleRequest);
        System.out.println("Successfully deleted the rule");
    }

    public static void deleteTargetsFromRule(EventBridgeClient eventBrClient,
String eventRuleName) {
        // First, get all targets that will be deleted.
        ListTargetsByRuleRequest request = ListTargetsByRuleRequest.builder()
                .rule(eventRuleName)
                .build();

        ListTargetsByRuleResponse response =
eventBrClient.listTargetsByRule(request);
        List<Target> allTargets = response.targets();

        // Get all targets and delete them.
        for (Target myTarget : allTargets) {
            RemoveTargetsRequest removeTargetsRequest =
RemoveTargetsRequest.builder()
                    .rule(eventRuleName)
                    .ids(myTarget.id())
                    .build();

            eventBrClient.removeTargets(removeTargetsRequest);
            System.out.println("Successfully removed the target");
        }
    }

    public static void triggerCustomRule(EventBridgeClient eventBrClient, String
email) {
        String json = "{" +
```

```
                "\"UserEmail\": \"" + email + "\"," +
                "\"Message\": \"This event was generated by example code.\"," +
                "\"UtcTime\": \"Now.\"" +
                "}";

        PutEventsRequestEntry entry = PutEventsRequestEntry.builder()
                .source("ExampleSource")
                .detail(json)
                .detailType("ExampleType")
                .build();

        PutEventsRequest eventsRequest = PutEventsRequest.builder()
                .entries(entry)
                .build();

        eventBrClient.putEvents(eventsRequest);
    }

    public static void updateCustomRuleTargetWithTransform(EventBridgeClient
eventBrClient, String topicArn,
            String ruleName) {
        String targetId = java.util.UUID.randomUUID().toString();
        InputTransformer inputTransformer = InputTransformer.builder()
                .inputTemplate("\"Notification: sample event was received.\"")
                .build();

        Target target = Target.builder()
                .id(targetId)
                .arn(topicArn)
                .inputTransformer(inputTransformer)
                .build();

        try {
            PutTargetsRequest targetsRequest = PutTargetsRequest.builder()
                    .rule(ruleName)
                    .targets(target)
                    .eventBusName(null)
                    .build();

            eventBrClient.putTargets(targetsRequest);
        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
```

```
    }

    public static void updateToCustomRule(EventBridgeClient eventBrClient, String
ruleName) {
        String customEventsPattern = "{" +
                "\"source\": [\"ExampleSource\"]," +
                "\"detail-type\": [\"ExampleType\"]" +
                "}";

        PutRuleRequest request = PutRuleRequest.builder()
                .name(ruleName)
                .description("Custom test rule")
                .eventPattern(customEventsPattern)
                .build();

        eventBrClient.putRule(request);
    }

    // Update an Amazon S3 object created rule with a transform on the target.
    public static void updateSnsEventRule(EventBridgeClient eventBrClient, String
topicArn, String ruleName) {
        String targetId = java.util.UUID.randomUUID().toString();
        Map<String, String> myMap = new HashMap<>();
        myMap.put("bucket", "$.detail.bucket.name");
        myMap.put("time", "$.time");

        InputTransformer inputTransformer = InputTransformer.builder()
                .inputTemplate("\"Notification: an object was uploaded to bucket
<bucket> at <time>.\"")
                .inputPathsMap(myMap)
                .build();

        Target target = Target.builder()
                .id(targetId)
                .arn(topicArn)
                .inputTransformer(inputTransformer)
                .build();

        try {
            PutTargetsRequest targetsRequest = PutTargetsRequest.builder()
                    .rule(ruleName)
                    .targets(target)
                    .eventBusName(null)
                    .build();
```

```
            eventBrClient.putTargets(targetsRequest);

        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void checkRule(EventBridgeClient eventBrClient, String
eventRuleName) {
        try {
            DescribeRuleRequest ruleRequest = DescribeRuleRequest.builder()
                    .name(eventRuleName)
                    .build();

            DescribeRuleResponse response =
eventBrClient.describeRule(ruleRequest);
            System.out.println("The state of the rule is " +
response.stateAsString());

        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void changeRuleState(EventBridgeClient eventBrClient, String
eventRuleName, Boolean isEnabled) {
        try {
            if (!isEnabled) {
                System.out.println("Disabling the rule: " + eventRuleName);
                DisableRuleRequest ruleRequest = DisableRuleRequest.builder()
                        .name(eventRuleName)
                        .build();

                eventBrClient.disableRule(ruleRequest);
            } else {
                System.out.println("Enabling the rule: " + eventRuleName);
                EnableRuleRequest ruleRequest = EnableRuleRequest.builder()
                        .name(eventRuleName)
                        .build();
                eventBrClient.enableRule(ruleRequest);
            }
```

```
        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Create and upload a file to an S3 bucket to trigger an event.
    public static void uploadTextFiletoS3(S3Client s3Client, String bucketName)
throws IOException {
        // Create a unique file name.
        String fileSuffix = new SimpleDateFormat("yyyyMMddHHmmss").format(new
Date());
        String fileName = "TextFile" + fileSuffix + ".txt";

        File myFile = new File(fileName);
        FileWriter fw = new FileWriter(myFile.getAbsoluteFile());
        BufferedWriter bw = new BufferedWriter(fw);
        bw.write("This is a sample file for testing uploads.");
        bw.close();

        try {
            PutObjectRequest putOb = PutObjectRequest.builder()
                    .bucket(bucketName)
                    .key(fileName)
                    .build();

            s3Client.putObject(putOb, RequestBody.fromFile(myFile));

        } catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void listTargetRules(EventBridgeClient eventBrClient, String
topicArn) {
        ListRuleNamesByTargetRequest ruleNamesByTargetRequest =
ListRuleNamesByTargetRequest.builder()
                .targetArn(topicArn)
                .build();

        ListRuleNamesByTargetResponse response =
eventBrClient.listRuleNamesByTarget(ruleNamesByTargetRequest);
```

```
        List<String> rules = response.ruleNames();
        for (String rule : rules) {
            System.out.println("The rule name is " + rule);
        }
    }

    public static void listTargets(EventBridgeClient eventBrClient, String
ruleName) {
        ListTargetsByRuleRequest ruleRequest = ListTargetsByRuleRequest.builder()
                .rule(ruleName)
                .build();

        ListTargetsByRuleResponse res =
eventBrClient.listTargetsByRule(ruleRequest);
        List<Target> targetsList = res.targets();
        for (Target target: targetsList) {
            System.out.println("Target ARN: "+target.arn());
        }
    }

    // Add a rule which triggers an SNS target when a file is uploaded to an S3
    // bucket.
    public static void addSnsEventRule(EventBridgeClient eventBrClient, String
ruleName, String topicArn,
            String topicName, String eventRuleName, String bucketName) {
        String targetID = java.util.UUID.randomUUID().toString();
        Target myTarget = Target.builder()
                .id(targetID)
                .arn(topicArn)
                .build();

        List<Target> targets = new ArrayList<>();
        targets.add(myTarget);
        PutTargetsRequest request = PutTargetsRequest.builder()
                .eventBusName(null)
                .targets(targets)
                .rule(ruleName)
                .build();

        eventBrClient.putTargets(request);
        System.out.println("Added event rule " + eventRuleName + " with Amazon
SNS target " + topicName + " for bucket "
                + bucketName + ".");
    }
```

```
   public static void subEmail(SnsClient snsClient, String topicArn, String
email) {
       try {
           SubscribeRequest request = SubscribeRequest.builder()
                   .protocol("email")
                   .endpoint(email)
                   .returnSubscriptionArn(true)
                   .topicArn(topicArn)
                   .build();

           SubscribeResponse result = snsClient.subscribe(request);
           System.out.println("Subscription ARN: " + result.subscriptionArn() +
"\n\n Status is "
                   + result.sdkHttpResponse().statusCode());

       } catch (SnsException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void listRules(EventBridgeClient eventBrClient) {
       try {
           ListRulesRequest rulesRequest = ListRulesRequest.builder()
                   .eventBusName("default")
                   .limit(10)
                   .build();

           ListRulesResponse response = eventBrClient.listRules(rulesRequest);
           List<Rule> rules = response.rules();
           for (Rule rule : rules) {
               System.out.println("The rule name is : " + rule.name());
               System.out.println("The rule description is : " +
rule.description());
               System.out.println("The rule state is : " +
rule.stateAsString());
           }

       } catch (EventBridgeException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

```
   public static String createSnsTopic(SnsClient snsClient, String topicName) {
       String topicPolicy = "{" +
               "\"Version\": \"2012-10-17\"," +
               "\"Statement\": [{" +
               "\"Sid\": \"EventBridgePublishTopic\"," +
               "\"Effect\": \"Allow\"," +
               "\"Principal\": {" +
               "\"Service\": \"events.amazonaws.com\"" +
               "}," +
               "\"Resource\": \"*\"," +
               "\"Action\": \"sns:Publish\"" +
               "}]" +
               "}";

       Map<String, String> topicAttributes = new HashMap<>();
       topicAttributes.put("Policy", topicPolicy);
       CreateTopicRequest topicRequest = CreateTopicRequest.builder()
               .name(topicName)
               .attributes(topicAttributes)
               .build();

       CreateTopicResponse response = snsClient.createTopic(topicRequest);
       System.out.println("Added topic " + topicName + " for email
subscriptions.");
       return response.topicArn();
   }

   // Create a new event rule that triggers when an Amazon S3 object is created
in
   // a bucket.
   public static void addEventRule(EventBridgeClient eventBrClient, String
roleArn, String bucketName,
           String eventRuleName) {
       String pattern = "{\n" +
               "  \"source\": [\"aws.s3\"],\n" +
               "  \"detail-type\": [\"Object Created\"],\n" +
               "  \"detail\": {\n" +
               "    \"bucket\": {\n" +
               "      \"name\": [\"" + bucketName + "\"]\n" +
               "    }\n" +
               "  }\n" +
               "}";
```

```
        try {
            PutRuleRequest ruleRequest = PutRuleRequest.builder()
                    .description("Created by using the AWS SDK for Java v2")
                    .name(eventRuleName)
                    .eventPattern(pattern)
                    .roleArn(roleArn)
                    .build();

            PutRuleResponse ruleResponse = eventBrClient.putRule(ruleRequest);
            System.out.println("The ARN of the new rule is " +
ruleResponse.ruleArn());

        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Determine if the S3 bucket exists.
    public static Boolean checkBucket(S3Client s3Client, String bucketName) {
        try {
            HeadBucketRequest headBucketRequest = HeadBucketRequest.builder()
                    .bucket(bucketName)
                    .build();

            s3Client.headBucket(headBucketRequest);
            return true;
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        return false;
    }

    // Set the S3 bucket notification configuration.
    public static void setBucketNotification(S3Client s3Client, String
bucketName) {
        try {
            EventBridgeConfiguration eventBridgeConfiguration =
EventBridgeConfiguration.builder()
                    .build();

            NotificationConfiguration configuration =
NotificationConfiguration.builder()
                    .eventBridgeConfiguration(eventBridgeConfiguration)
```

```
                    .build();

            PutBucketNotificationConfigurationRequest configurationRequest =
PutBucketNotificationConfigurationRequest
                    .builder()
                    .bucket(bucketName)
                    .notificationConfiguration(configuration)
                    .skipDestinationValidation(true)
                    .build();

            s3Client.putBucketNotificationConfiguration(configurationRequest);
            System.out.println("Added bucket " + bucketName + " with EventBridge
events enabled.");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void createBucket(S3Client s3Client, String bucketName) {
        try {
            S3Waiter s3Waiter = s3Client.waiter();
            CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
                    .bucket(bucketName)
                    .build();

            s3Client.createBucket(bucketRequest);
            HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
                    .bucket(bucketName)
                    .build();

            // Wait until the bucket is created and print out the response.
            WaiterResponse<HeadBucketResponse> waiterResponse =
s3Waiter.waitUntilBucketExists(bucketRequestWait);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println(bucketName + " is ready");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

```
    public static String createIAMRole(IamClient iam, String rolename, String
 polJSON) {
        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(polJSON)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            AttachRolePolicyRequest rolePolicyRequest =
 AttachRolePolicyRequest.builder()
                    .roleName(rolename)
                    .policyArn("arn:aws:iam::aws:policy/
AmazonEventBridgeFullAccess")
                    .build();

            iam.attachRolePolicy(rolePolicyRequest);
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for Java 2.x .
  - DeleteRule
  - DescribeRule
  - DisableRule
  - EnableRule
  - ListRuleNamesByTarget
  - ListRules
  - ListTargetsByRule
  - PutEvents
  - PutRule

- PutTargets

Kotlin

SDK for Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
/*
 Before running this Kotlin code example, set up your development environment,
 including your credentials.

 For more information, see the following documentation topic:
 https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html

 This Kotlin example performs the following tasks with Amazon EventBridge:

 1. Creates an AWS Identity and Access Management (IAM) role to use with Amazon
 EventBridge.
 2. Creates an Amazon Simple Storage Service (Amazon S3) bucket with EventBridge
 events enabled.
 3. Creates a rule that triggers when an object is uploaded to Amazon S3.
 4. Lists rules on the event bus.
 5. Creates a new Amazon Simple Notification Service (Amazon SNS) topic and lets
 the user subscribe to it.
 6. Adds a target to the rule that sends an email to the specified topic.
 7. Creates an EventBridge event that sends an email when an Amazon S3 object is
 created.
 8. Lists targets.
 9. Lists the rules for the same target.
 10. Triggers the rule by uploading a file to the S3 bucket.
 11. Disables a specific rule.
 12. Checks and prints the state of the rule.
 13. Adds a transform to the rule to change the text of the email.
 14. Enables a specific rule.
 15. Triggers the updated rule by uploading a file to the S3 bucket.
 16. Updates the rule to a custom rule pattern.
```

```
 17. Sends an event to trigger the rule.
 18. Cleans up resources.
*/
val DASHES: String = String(CharArray(80)).replace("\u0000", "-")
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <roleName> <bucketName> <topicName> <eventRuleName>

    Where:
        roleName - The name of the role to create.
        bucketName - The Amazon Simple Storage Service (Amazon S3) bucket name to
 create.
        topicName - The name of the Amazon Simple Notification Service (Amazon
 SNS) topic to create.
        eventRuleName - The Amazon EventBridge rule name to create.
    """
    val polJSON = "{" +
        "\"Version\": \"2012-10-17\"," +
        "\"Statement\": [{" +
        "\"Effect\": \"Allow\"," +
        "\"Principal\": {" +
        "\"Service\": \"events.amazonaws.com\"" +
        "}," +
        "\"Action\": \"sts:AssumeRole\"" +
        "}]" +
        "}"

    if (args.size != 4) {
        println(usage)
        exitProcess(1)
    }

    val sc = Scanner(System.`in`)
    val roleName = args[0]
    val bucketName = args[1]
    val topicName = args[2]
    val eventRuleName = args[3]

    println(DASHES)
    println("Welcome to the Amazon EventBridge example scenario.")
    println(DASHES)

    println(DASHES)
```

```
    println("1. Create an AWS Identity and Access Management (IAM) role to use
with Amazon EventBridge.")
    val roleArn = createIAMRole(roleName, polJSON)
    println(DASHES)

    println(DASHES)
    println("2. Create an S3 bucket with EventBridge events enabled.")
    if (checkBucket(bucketName)) {
        println("$bucketName already exists. Ending this scenario.")
        exitProcess(1)
    }

    createBucket(bucketName)
    delay(3000)
    setBucketNotification(bucketName)
    println(DASHES)

    println(DASHES)
    println("3. Create a rule that triggers when an object is uploaded to Amazon
S3.")
    delay(10000)
    addEventRule(roleArn, bucketName, eventRuleName)
    println(DASHES)

    println(DASHES)
    println("4. List rules on the event bus.")
    listRules()
    println(DASHES)

    println(DASHES)
    println("5. Create a new SNS topic for testing and let the user subscribe to
the topic.")
    val topicArn = createSnsTopic(topicName)
    println(DASHES)

    println(DASHES)
    println("6. Add a target to the rule that sends an email to the specified
topic.")
    println("Enter your email to subscribe to the Amazon SNS topic:")
    val email = sc.nextLine()
    subEmail(topicArn, email)
    println("Use the link in the email you received to confirm your subscription.
Then press Enter to continue.")
    sc.nextLine()
```

```
    println(DASHES)

    println(DASHES)
    println("7. Create an EventBridge event that sends an email when an Amazon S3
object is created.")
    addSnsEventRule(eventRuleName, topicArn, topicName, eventRuleName,
bucketName)
    println(DASHES)

    println(DASHES)
    println("8. List targets.")
    listTargets(eventRuleName)
    println(DASHES)

    println(DASHES)
    println(" 9. List the rules for the same target.")
    listTargetRules(topicArn)
    println(DASHES)

    println(DASHES)
    println("10. Trigger the rule by uploading a file to the S3 bucket.")
    println("Press Enter to continue.")
    sc.nextLine()
    uploadTextFiletoS3(bucketName)
    println(DASHES)

    println(DASHES)
    println("11. Disable a specific rule.")
    changeRuleState(eventRuleName, false)
    println(DASHES)

    println(DASHES)
    println("12. Check and print the state of the rule.")
    checkRule(eventRuleName)
    println(DASHES)

    println(DASHES)
    println("13. Add a transform to the rule to change the text of the email.")
    updateSnsEventRule(topicArn, eventRuleName)
    println(DASHES)

    println(DASHES)
    println("14. Enable a specific rule.")
    changeRuleState(eventRuleName, true)
```

```
    println(DASHES)

    println(DASHES)
    println("15. Trigger the updated rule by uploading a file to the S3 bucket.")
    println("Press Enter to continue.")
    sc.nextLine()
    uploadTextFiletoS3(bucketName)
    println(DASHES)

    println(DASHES)
    println("16. Update the rule to a custom rule pattern.")
    updateToCustomRule(eventRuleName)
    println("Updated event rule $eventRuleName to use a custom pattern.")
    updateCustomRuleTargetWithTransform(topicArn, eventRuleName)
    println("Updated event target $topicArn.")
    println(DASHES)

    println(DASHES)
    println("17. Send an event to trigger the rule. This will trigger a
 subscription email.")
    triggerCustomRule(email)
    println("Events have been sent. Press Enter to continue.")
    sc.nextLine()
    println(DASHES)

    println(DASHES)
    println("18. Clean up resources.")
    println("Do you want to clean up resources (y/n)")
    val ans = sc.nextLine()
    if (ans.compareTo("y") == 0) {
        cleanupResources(topicArn, eventRuleName, bucketName, roleName)
    } else {
        println("The resources will not be cleaned up. ")
    }
    println(DASHES)

    println(DASHES)
    println("The Amazon EventBridge example scenario has successfully
 completed.")
    println(DASHES)
}

suspend fun cleanupResources(topicArn: String?, eventRuleName: String?,
 bucketName: String?, roleName: String?) {
```

```
    println("Removing all targets from the event rule.")
    deleteTargetsFromRule(eventRuleName)
    deleteRuleByName(eventRuleName)
    deleteSNSTopic(topicArn)
    deleteS3Bucket(bucketName)
    deleteRole(roleName)
}

suspend fun deleteRole(roleNameVal: String?) {
    val policyArnVal = "arn:aws:iam::aws:policy/AmazonEventBridgeFullAccess"
    val policyRequest = DetachRolePolicyRequest {
        policyArn = policyArnVal
        roleName = roleNameVal
    }
    IamClient { region = "us-east-1" }.use { iam ->
        iam.detachRolePolicy(policyRequest)
        println("Successfully detached policy $policyArnVal from role
 $roleNameVal")

        // Delete the role.
        val roleRequest = DeleteRoleRequest {
            roleName = roleNameVal
        }

        iam.deleteRole(roleRequest)
        println("*** Successfully deleted $roleNameVal")
    }
}

suspend fun deleteS3Bucket(bucketName: String?) {
    // Remove all the objects from the S3 bucket.
    val listObjects = ListObjectsRequest {
        bucket = bucketName
    }
    S3Client { region = "us-east-1" }.use { s3Client ->
        val res = s3Client.listObjects(listObjects)
        val myObjects = res.contents
        val toDelete = mutableListOf<ObjectIdentifier>()

        if (myObjects != null) {
            for (myValue in myObjects) {
                toDelete.add(
                    ObjectIdentifier {
                        key = myValue.key
```

```
                    }
                )
            }
        }

        val delOb = Delete {
            objects = toDelete
        }

        val dor = DeleteObjectsRequest {
            bucket = bucketName
            delete = delOb
        }
        s3Client.deleteObjects(dor)

        // Delete the S3 bucket.
        val deleteBucketRequest = DeleteBucketRequest {
            bucket = bucketName
        }
        s3Client.deleteBucket(deleteBucketRequest)
        println("You have deleted the bucket and the objects")
    }
}

// Delete the SNS topic.
suspend fun deleteSNSTopic(topicArnVal: String?) {
    val request = DeleteTopicRequest {
        topicArn = topicArnVal
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        snsClient.deleteTopic(request)
        println(" $topicArnVal was deleted.")
    }
}

suspend fun deleteRuleByName(ruleName: String?) {
    val ruleRequest = DeleteRuleRequest {
        name = ruleName
    }
    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.deleteRule(ruleRequest)
        println("Successfully deleted the rule")
    }
```

```
}

suspend fun deleteTargetsFromRule(eventRuleName: String?) {
    // First, get all targets that will be deleted.
    val request = ListTargetsByRuleRequest {
        rule = eventRuleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.listTargetsByRule(request)
        val allTargets = response.targets

        // Get all targets and delete them.
        if (allTargets != null) {
            for (myTarget in allTargets) {
                val removeTargetsRequest = RemoveTargetsRequest {
                    rule = eventRuleName
                    ids = listOf(myTarget.id.toString())
                }
                eventBrClient.removeTargets(removeTargetsRequest)
                println("Successfully removed the target")
            }
        }
    }
}

suspend fun triggerCustomRule(email: String) {
    val json = "{" +
        "\"UserEmail\": \"" + email + "\"," +
        "\"Message\": \"This event was generated by example code.\"" +
        "\"UtcTime\": \"Now.\"" +
        "}"

    val entry = PutEventsRequestEntry {
        source = "ExampleSource"
        detail = json
        detailType = "ExampleType"
    }

    val eventsRequest = PutEventsRequest {
        this.entries = listOf(entry)
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
```

```
        eventBrClient.putEvents(eventsRequest)
    }
}

suspend fun updateCustomRuleTargetWithTransform(topicArn: String?, ruleName:
 String?) {
    val targetId = UUID.randomUUID().toString()

    val inputTransformerOb = InputTransformer {
        inputTemplate = "\"Notification: sample event was received.\""
    }

    val target = Target {
        id = targetId
        arn = topicArn
        inputTransformer = inputTransformerOb
    }

    val targetsRequest = PutTargetsRequest {
        rule = ruleName
        targets = listOf(target)
        eventBusName = null
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putTargets(targetsRequest)
    }
}

suspend fun updateToCustomRule(ruleName: String?) {
    val customEventsPattern = "{" +
        "\"source\": [\"ExampleSource\"]," +
        "\"detail-type\": [\"ExampleType\"]" +
        "}"
    val request = PutRuleRequest {
        name = ruleName
        description = "Custom test rule"
        eventPattern = customEventsPattern
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putRule(request)
    }
}
```

```
// Update an Amazon S3 object created rule with a transform on the target.
suspend fun updateSnsEventRule(topicArn: String?, ruleName: String?) {
    val targetId = UUID.randomUUID().toString()
    val myMap = mutableMapOf<String, String>()
    myMap["bucket"] = "$.detail.bucket.name"
    myMap["time"] = "$.time"

    val inputTransOb = InputTransformer {
        inputTemplate = "\"Notification: an object was uploaded to bucket
 <bucket> at <time>.\""
        inputPathsMap = myMap
    }
    val targetOb = Target {
        id = targetId
        arn = topicArn
        inputTransformer = inputTransOb
    }

    val targetsRequest = PutTargetsRequest {
        rule = ruleName
        targets = listOf(targetOb)
        eventBusName = null
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putTargets(targetsRequest)
    }
}

suspend fun checkRule(eventRuleName: String?) {
    val ruleRequest = DescribeRuleRequest {
        name = eventRuleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.describeRule(ruleRequest)
        println("The state of the rule is $response")
    }
}

suspend fun changeRuleState(eventRuleName: String, isEnabled: Boolean?) {
    if (!isEnabled!!) {
        println("Disabling the rule: $eventRuleName")
```

```
        val ruleRequest = DisableRuleRequest {
            name = eventRuleName
        }
        EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
            eventBrClient.disableRule(ruleRequest)
        }
    } else {
        println("Enabling the rule: $eventRuleName")
        val ruleRequest = EnableRuleRequest {
            name = eventRuleName
        }
        EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
            eventBrClient.enableRule(ruleRequest)
        }
    }
}

// Create and upload a file to an S3 bucket to trigger an event.
@Throws(IOException::class)
suspend fun uploadTextFiletoS3(bucketName: String?) {
    val fileSuffix = SimpleDateFormat("yyyyMMddHHmmss").format(Date())
    val fileName = "TextFile$fileSuffix.txt"
    val myFile = File(fileName)
    val fw = FileWriter(myFile.absoluteFile)
    val bw = BufferedWriter(fw)
    bw.write("This is a sample file for testing uploads.")
    bw.close()

    val putOb = PutObjectRequest {
        bucket = bucketName
        key = fileName
        body = myFile.asByteStream()
    }

    S3Client { region = "us-east-1" }.use { s3Client ->
        s3Client.putObject(putOb)
    }
}

suspend fun listTargetRules(topicArnVal: String?) {
    val ruleNamesByTargetRequest = ListRuleNamesByTargetRequest {
        targetArn = topicArnVal
    }
```

```
    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response =
 eventBrClient.listRuleNamesByTarget(ruleNamesByTargetRequest)
        response.ruleNames?.forEach { rule ->
            println("The rule name is $rule")
        }
    }
}

suspend fun listTargets(ruleName: String?) {
    val ruleRequest = ListTargetsByRuleRequest {
        rule = ruleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.listTargetsByRule(ruleRequest)
        response.targets?.forEach { target ->
            println("Target ARN: ${target.arn}")
        }
    }
}

// Add a rule that triggers an SNS target when a file is uploaded to an S3
 bucket.
suspend fun addSnsEventRule(ruleName: String?, topicArn: String?, topicName:
 String, eventRuleName: String, bucketName: String) {
    val targetID = UUID.randomUUID().toString()
    val myTarget = Target {
        id = targetID
        arn = topicArn
    }

    val targetsOb = mutableListOf<Target>()
    targetsOb.add(myTarget)

    val request = PutTargetsRequest {
        eventBusName = null
        targets = targetsOb
        rule = ruleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putTargets(request)
```

```
        println("Added event rule $eventRuleName with Amazon SNS target
 $topicName for bucket $bucketName.")
    }
}

suspend fun subEmail(topicArnVal: String?, email: String?) {
    val request = SubscribeRequest {
        protocol = "email"
        endpoint = email
        returnSubscriptionArn = true
        topicArn = topicArnVal
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        val result = snsClient.subscribe(request)
        println(" Subscription ARN: ${result.subscriptionArn}")
    }
}

suspend fun createSnsTopic(topicName: String): String? {
    val topicPolicy = "{" +
        "\"Version\": \"2012-10-17\"," +
        "\"Statement\": [{" +
        "\"Sid\": \"EventBridgePublishTopic\"," +
        "\"Effect\": \"Allow\"," +
        "\"Principal\": {" +
        "\"Service\": \"events.amazonaws.com\"" +
        "}," +
        "\"Resource\": \"*\"," +
        "\"Action\": \"sns:Publish\"" +
        "}]" +
        "}"

    val topicAttributes = mutableMapOf<String, String>()
    topicAttributes["Policy"] = topicPolicy

    val topicRequest = CreateTopicRequest {
        name = topicName
        attributes = topicAttributes
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        val response = snsClient.createTopic(topicRequest)
        println("Added topic $topicName for email subscriptions.")
```

```
        return response.topicArn
    }
}

suspend fun listRules() {
    val rulesRequest = ListRulesRequest {
        eventBusName = "default"
        limit = 10
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.listRules(rulesRequest)
        response.rules?.forEach { rule ->
            println("The rule name is ${rule.name}")
            println("The rule ARN is ${rule.arn}")
        }
    }
}

// Create a new event rule that triggers when an Amazon S3 object is created in a
 bucket.
suspend fun addEventRule(roleArnVal: String?, bucketName: String, eventRuleName:
 String?) {
    val pattern = """{
        "source": ["aws.s3"],
        "detail-type": ["Object Created"],
        "detail": {
        "bucket": {
            "name": ["$bucketName"]
            }
        }
    }"""

    val ruleRequest = PutRuleRequest {
        description = "Created by using the AWS SDK for Kotlin"
        name = eventRuleName
        eventPattern = pattern
        roleArn = roleArnVal
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val ruleResponse = eventBrClient.putRule(ruleRequest)
        println("The ARN of the new rule is ${ruleResponse.ruleArn}")
    }
```

```
}

// Set the Amazon S3 bucket notification configuration.
suspend fun setBucketNotification(bucketName: String) {
    val eventBridgeConfig = EventBridgeConfiguration {
    }

    val configuration = NotificationConfiguration {
        eventBridgeConfiguration = eventBridgeConfig
    }

    val configurationRequest = PutBucketNotificationConfigurationRequest {
        bucket = bucketName
        notificationConfiguration = configuration
        skipDestinationValidation = true
    }

    S3Client { region = "us-east-1" }.use { s3Client ->
        s3Client.putBucketNotificationConfiguration(configurationRequest)
        println("Added bucket $bucketName with EventBridge events enabled.")
    }
}

// Create an S3 bucket using a waiter.
suspend fun createBucket(bucketName: String) {
    val request = CreateBucketRequest {
        bucket = bucketName
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.createBucket(request)
        s3.waitUntilBucketExists {
            bucket = bucketName
        }
        println("$bucketName is ready")
    }
}

suspend fun checkBucket(bucketName: String?): Boolean {
    try {
        // Determine if the S3 bucket exists.
        val headBucketRequest = HeadBucketRequest {
            bucket = bucketName
        }
```

```
        S3Client { region = "us-east-1" }.use { s3Client ->
            s3Client.headBucket(headBucketRequest)
            return true
        }
    } catch (e: S3Exception) {
        System.err.println(e.message)
    }
    return false
}

suspend fun createIAMRole(rolenameVal: String?, polJSON: String?): String? {
    val request = CreateRoleRequest {
        roleName = rolenameVal
        assumeRolePolicyDocument = polJSON
        description = "Created using the AWS SDK for Kotlin"
    }

    val rolePolicyRequest = AttachRolePolicyRequest {
        roleName = rolenameVal
        policyArn = "arn:aws:iam::aws:policy/AmazonEventBridgeFullAccess"
    }

    IamClient { region = "us-east-1" }.use { iam ->
        val response = iam.createRole(request)
        iam.attachRolePolicy(rolePolicyRequest)
        return response.role?.arn
    }
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK For Kotlin.
  - DeleteRule
  - DescribeRule
  - DisableRule
  - EnableRule
  - ListRuleNamesByTarget
  - ListRules
  - ListTargetsByRule
  - PutEvents

- PutRule
- PutTargets

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Contoh lintas layanan untuk EventBridge menggunakan AWS SDK

Contoh aplikasi berikut menggunakan AWS SDK untuk digabungkan EventBridge dengan yang lain Layanan AWS. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan petunjuk tentang cara mengatur dan menjalankan aplikasi.

Contoh-contoh

- Menggunakan peristiwa terjadwal untuk menginvokasi fungsi Lambda

Menggunakan peristiwa terjadwal untuk menginvokasi fungsi Lambda

Contoh kode berikut menunjukkan cara membuat AWS Lambda fungsi yang dipanggil oleh acara EventBridge terjadwal Amazon.

Java

SDK for Java 2.x

Menunjukkan cara membuat acara EventBridge terjadwal Amazon yang memanggil AWS Lambda fungsi. Konfigurasikan EventBridge untuk menggunakan ekspresi cron untuk menjadwalkan saat fungsi Lambda dipanggil. Dalam contoh ini, Anda membuat fungsi Lambda menggunakan API runtime Java Lambda. Contoh ini memanggil AWS layanan yang berbeda untuk melakukan kasus penggunaan tertentu. Contoh ini menunjukkan cara membuat aplikasi yang mengirimkan pesan teks seluler kepada karyawan Anda berisi ucapan selamat pada hari jadi setahun kerja mereka.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- DynamoDB

- EventBridge
- Lambda
- Amazon SNS

JavaScript

SDK untuk JavaScript (v3)

Menunjukkan cara membuat acara EventBridge terjadwal Amazon yang memanggil AWS Lambda fungsi. Konfigurasikan EventBridge untuk menggunakan ekspresi cron untuk menjadwalkan saat fungsi Lambda dipanggil. Dalam contoh ini, Anda membuat fungsi Lambda menggunakan API runtime JavaScript Lambda. Contoh ini memanggil AWS layanan yang berbeda untuk melakukan kasus penggunaan tertentu. Contoh ini menunjukkan cara membuat aplikasi yang mengirimkan pesan teks seluler kepada karyawan Anda berisi ucapan selamat pada hari jadi setahun kerja mereka.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Contoh ini juga tersedia di panduan developer AWS SDK for JavaScript v3.

Layanan yang digunakan dalam contoh ini

- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

Python

SDK for Python (Boto3)

Contoh ini menunjukkan cara mendaftarkan AWS Lambda fungsi sebagai target EventBridge acara Amazon terjadwal. Penangan Lambda menulis pesan ramah dan data peristiwa lengkap ke Amazon CloudWatch Logs untuk pengambilan nanti.

- Menyebarkan fungsi Lambda.
- Membuat acara EventBridge terjadwal dan menjadikan fungsi Lambda sebagai target.
- Memberikan izin untuk membiarkan EventBridge menjalankan fungsi Lambda.

- Mencetak data terbaru dari CloudWatch Log untuk menampilkan hasil pemanggilan terjadwal.
- Membersihkan semua sumber daya yang dibuat selama demo.

Contoh ini paling baik dilihat di GitHub. Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- CloudWatch Log
- EventBridge
- Lambda

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan EventBridge dengan AWS SDK. Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

EventBridge Keamanan Amazon

Amazon EventBridge menggunakan AWS Identity and Access Management untuk mengontrol akses ke AWS layanan dan sumber daya lain. Untuk gambaran umum mengenai cara kerja IAM, lihat Gambaran Umum Manajemen Akses di Panduan Pengguna IAM. Untuk ikhtisar kredenal keamanan, lihat Kredensial AWS Keamanan di. Referensi Umum Amazon Web

Topik

- Perlindungan data di Amazon EventBridge
- Kebijakan Berbasis Tanda
- Amazon EventBridge dan AWS Identity and Access Management
- Logging panggilan Amazon EventBridge API menggunakan AWS CloudTrail
- Validasi kepatuhan di Amazon EventBridge
- Ketahanan Amazon EventBridge
- Keamanan infrastruktur di AmazonEventBridge
- Analisis konfigurasi dan kelemahan di Amazon EventBridge (Amazon EventBridge)

Perlindungan data di Amazon EventBridge

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data di Amazon EventBridge. Sebagaimana diuraikan dalam model ini, AWS bertanggung jawab untuk memberikan perlindungan terhadap infrastruktur global yang menjalankan semua AWS Cloud. Anda harus bertanggung jawab untuk memelihara kendali terhadap konten yang di-hosting pada infrastruktur ini. Anda juga bertanggung jawab atas konfigurasi keamanan dan tugas manajemen untuk berbagai layanan Layanan AWS yang Anda gunakan. Untuk informasi selengkapnya tentang privasi data, silakan lihat Pertanyaan Umum Privasi Data. Untuk informasi tentang perlindungan data di Eropa, silakan lihat postingan blog Model Tanggung Jawab Bersama AWS dan GDPR di Blog Keamanan AWS.

Untuk tujuan perlindungan data, sebaiknya Anda melindungi kredensial Akun AWS dan menyiapkan akun pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara seperti itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugas mereka. Kami juga merekomendasikan agar Anda mengamankan data Anda dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk melakukan komunikasi dengan sumber daya AWS. Kami mensyaratkan TLS 1.2 dan merekomendasikan TLS 1.3.
- Siapkan API dan log aktivitas pengguna dengan AWS CloudTrail.
- Gunakan solusi enkripsi AWS, bersama dengan semua kontrol keamanan default dalam layanan Layanan AWS.
- Gunakan layanan keamanan terkelola lanjutan seperti Amazon Macie, yang membantu menemukan dan mengamankan data pribadi yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-2 ketika mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Untuk informasi selengkapnya tentang titik akhir FIPS yang tersedia, silakan lihat Standar Pemrosesan Informasi Federal (FIPS) 140-2.

Sebaiknya Anda tidak memasukkan informasi rahasia atau sensitif, seperti alamat email pelanggan, ke dalam tanda atau bidang teks bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan EventBridge atau lainnya Layanan AWS menggunakan konsol, APIAWS CLI, atau AWS SDK. Data apa pun yang Anda masukkan ke dalam tanda atau bidang teks bebas yang digunakan untuk nama dapat digunakan untuk log penagihan atau diagnostik. Saat Anda memberikan URL ke

server eksternal, kami sangat menyarankan jangan menyertakan informasi kredensial di URL untuk memvalidasi permintaan Anda ke server tersebut.

Enkripsi saat tidak aktif

EventBridge mengenkripsi metadata peristiwa dan data pesan yang disimpannya. Secara default, EventBridge mengenkripsi data menggunakan 256-bit Advanced Encryption Standard (AES-256) di bawah kunci yang AWS dimiliki, yang membantu mengamankan data Anda dari akses yang tidak sah. Tidak ada biaya tambahan untuk mengenkripsi data Anda dengan menggunakan kunci yang AWS dimiliki.

Enkripsi dalam transit

EventBridge mengenkripsi data yang melewati antara EventBridge dan layanan lainnya dengan menggunakan Transport layer Security (TLS).

Kebijakan Berbasis Tanda

Di Amazon EventBridge, Anda dapat menggunakan kebijakan berdasarkan tag untuk mengontrol akses ke sumber daya.

Misalnya, Anda dapat membatasi akses ke sumber daya yang menyertakan tanda dengan kunci `environment` dan nilai `production`. Contoh kebijakan berikut menolak semua sumber daya dengan tanda ini kemampuan untuk membuat, menghapus, atau mengubah tanda, aturan, atau bus peristiwa untuk sumber daya yang telah diberi tanda `environment/production`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "events:PutRule",
                "events:DescribeRule",
                "events:DeleteRule",
                "events:CreateEventBus",
                "events:DescribeEventBus"
                "events:DeleteEventBus"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/environment": "production"}
            }
        }
    ]
}
```

Untuk informasi lebih lanjut tentang penandaan, lihat berikut ini.

- EventBridge Tag Amazon
- Mengontrol Akses Menggunakan Tag IAM

Amazon EventBridge dan AWS Identity and Access Management

Untuk mengakses Amazon EventBridge, Anda memerlukan kredensi yang AWS dapat digunakan untuk mengautentikasi permintaan Anda. Kredensial tersebut harus memiliki izin untuk mengakses AWS sumber daya, seperti mengambil data peristiwa dari sumber daya AWS lain. Bagian berikut memberikan rincian tentang bagaimana Anda dapat menggunakan AWS Identity and Access Management(IAM) dan EventBridge untuk membantu mengamankan sumber daya Anda dengan mengontrol siapa yang dapat mengaksesnya.

Topik

- Autentikasi
- Kontrol akses
- Mengelola izin akses keEventBridge sumber daya Amazon Anda
- Menggunakan kebijakan berbasis identitas (kebijakan IAM) untuk Amazon EventBridge
- Menggunakan kebijakan berbasis sumber daya untuk AmazonEventBridge
- Pencegahan Deputi Bingung Lintas Layanan
- Kebijakan berbasis sumber daya untuk skema Amazon EventBridge
- Referensi izin Amazon EventBridge
- Menggunakan ketentuan kebijakan IAM ketentuan untuk kontrol akses yang sangat baik
- Menggunakan peran terkait layanan untuk EventBridge

Autentikasi

Anda dapat mengakses AWS sebagai salah satu tipe identitas berikut:

- Pengguna root akun AWS – Ketika mendaftar ke AWS, Anda memberikan alamat email dan kata sandi yang terkait dengan akun Anda. Ini adalah kredensial root Anda, dan mereka memberikan akses penuh ke semua sumber daya AWS Anda.

> **Important**
>
> Demi alasan keamanan, kami merekomendasikan agar Anda menggunakan kredensial root hanya untuk membuat administrator, yang merupakan Pengguna IAM dengan izin penuh untuk akun Anda. Kemudian Anda dapat menggunakan administrator ini untuk membuat pengguna dan peran lain dengan izin terbatas. Untuk informasi selengkapnya,

lihat Praktik Terbaik IAM dan Membuat Pengguna Admin dan Grup dalam Panduan Pengguna IAM.

- Pengguna IAM — Pengguna IAM adalah identitas dalam akun Anda yang memiliki izin khusus, misalnya, izin untuk mengirim data peristiwa ke target di. EventBridge Anda dapat menggunakan kredenal masuk IAM untuk masuk untuk mengamankan AWS halaman web seperti, Forum AWS Diskusi AWS Management Console, atau Pusat. AWS Support

  Selain kredensi masuk, Anda juga dapat membuat kunci akses untuk setiap pengguna. Anda dapat menggunakan kunci ini saat Anda mengakses AWS layanan secara programatik untuk menandatangani permintaan Anda secara kriptografis, baik melalui salah satu SDK atau dengan menggunakanAWS Command Line Interface (AWS CLI). Jika Anda tidak menggunakan alat AWS, Anda harus menandatangani permintaan tersebut sendiri denganVersi Tanda Tangan 4, protokol untuk mengautentikasi permintaan API inbound. Untuk informasi selengkapnya tentang melakukan autentikasi permintaan, lihat Proses Penandatanganan Tanda Tangan Versi 4 dalam Referensi Umum Amazon Web.

- IAM role – IAM role adalah identitas IAM yang dapat Anda buat di akun Anda yang memiliki izin spesifik. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. IAM role memungkinkan Anda memperoleh kunci akses sementara yang dapat digunakan untuk mengakses layanan dan sumber daya AWS. Peran IAM dengan kredensial temporer berguna dalam situasi berikut:

  - Akses pengguna federasi — Alih-alih membuat pengguna, Anda dapat menggunakan identitas dariAWS Directory Service, direktori pengguna perusahaan Anda, atau penyedia identitas web (iDP). Ini dikenal sebagaipengguna gabungan. AWSmenetapkan peran ke pengguna gabungan saat pengguna meminta akses melalui penyedia identitas. Untuk informasi lebih lanjut tentang pengguna federasi, lihat Pengguna dan Peran Gabungan dalam Panduan Pengguna IAM.
  - Akses lintas akun – Anda dapat menggunakan IAM role di akun Anda untuk memberikan izin akun lain untuk mengakses sumber daya akun Anda. Sebagai contoh, lihat Tutorial: Mendelegasikan Akses di Seluruh AWS Menggunakan Peran IAM dalam Panduan Pengguna IAM.
  - AWS akses layanan – Anda dapat menggunakan IAM role di akun Anda untuk memberikan izin layanan AWS untuk mengakses sumber daya akun Anda. Misalnya, Anda dapat membuat peran yang memungkinkan Amazon Redshift untuk memuat data yang disimpan di bucket Amazon S3 ke dalam klaster Amazon Redshift. Untuk informasi selengkapnya, lihat Membuat Peran untuk Mendelegasikan Izin ke AWS Layanan dalam Panduan Pengguna IAM.

- Aplikasi yang berjalan di Amazon EC2 — Untuk aplikasi Amazon EC2 yang memerlukan akses EventBridge, Anda dapat menyimpan kunci akses di instans EC2 atau Anda dapat menggunakan peran IAM untuk mengelola kredensi sementara. Untuk menetapkan peran AWS ke instans EC2, Anda membuat profil instans yang dilampirkan ke instans. Profil instans memuat peran dan memberikan kredensial sementara ke aplikasi yang berjalan di instans EC2. Untuk informasi selengkapnya, lihat Menggunakan Peran untuk Aplikasi di Amazon EC2 dalam Panduan Pengguna IAM.

Kontrol akses

Untuk membuat atau mengakses EventBridge sumber daya, Anda memerlukan kredensi dan izin yang valid. Sebagai contoh, untuk memohon AWS Lambda, Amazon Simple Notification Service (Amazon SNS), dan target Amazon Simple Queue Service (Amazon SQS), Anda harus memiliki izin untuk layanan tersebut.

Mengelola izin akses keEventBridge sumber daya Amazon Anda

Anda mengelola akses keEventBridge sumber daya seperti aturan atau peristiwa dengan menggunakan kebijakan berbasis identitas atau berbasis sumber daya.

Sumber daya EventBridge

EventBridgesumber daya dan subsumber daya memiliki Amazon Resource Names (ARN) unik yang terkait dengannya. Anda menggunakan ARN diEventBridge untuk membuat pola peristiwa. Untuk informasi selengkapnya tentang ARN, lihat Amazon Resource Names (ARN) dan NamespaceAWS Layanan di Ruang Nama Layanan dalam Referensi Umum Amazon Web Services.

Untuk daftar operasiEventBridge menyediakan untuk bekerja dengan sumber daya, lihatReferensi izin Amazon EventBridge.

> **Note**
>
> Sebagian besar layanan di AWSmenganggap tanda titik dua (`:`) atau garis miring (`/`) sebagai karakter yang sama di ARN. Namun,EventBridge menggunakan pencarian yang sama persis di pola peristiwa dan aturan. Pastikan untuk menggunakan karakter ARN yang benar saat membuat pola peristiwa sehingga mereka cocok dengan sintaks ARN dalam peristiwa yang Anda ingin cocokkan.

Tabel berikut menunjukkan sumber daya diEventBridge.

Jenis Sumber Daya	Format ARN
Arsip	`arn:aws:events:` *`region`*`:`*`account`*`:archive/` *`archive-name`*
Putar Ulang	`arn:aws:events:` *`region`*`:`*`account`*`:replay/`*`replay-name`
Aturan	`arn:aws:events:` *`region`*`:`*`account`*`:rule/`*`[event-bus-name]/rule-name`
Bus peristiwa	`arn:aws:events:` *`region`*`:`*`account`*`:event-bus/` *`event-bus-name`*

Jenis Sumber Daya	Format ARN
Semua sumber daya EventBridge	`arn:aws:events:*`
SemuaEventBridge sumber daya yang dimiliki oleh akun yang ditentukan di Wilayah yang ditentukan	`arn:aws:events: region:account:*`

Contoh berikut menunjukkan cara menunjukkan aturan tertentu (*myRule*) dalam pernyataan Anda menggunakan ARN-nya.

```
"Resource": "arn:aws:events:us-east-1:123456789012:rule/myRule"
```

Untuk menentukan semua aturan yang menjadi bagian dari akun tertentu dengan menggunakan tanda bintang (*) seperti berikut ini.

```
"Resource": "arn:aws:events:us-east-1:123456789012:rule/*"
```

Untuk menentukan semua sumber daya, atau jika tindakan API tertentu tidak mendukung ARN, gunakan tanda bintang (*) dalam elemen `Resource` seperti berikut ini.

```
"Resource": "*"
```

Untuk menentukan beberapa sumber daya atau `PutTargets` dalam satu pernyataan, pisahkan ARN dengan koma seperti berikut ini

```
"Resource": ["arn1", "arn2"]
```

Kepemilikan sumber daya

Akun memiliki sumber daya di akun, tidak peduli siapa yang membuat sumber daya. Pemilik sumber daya adalah akun dari entitas utama, pengguna akar akun, pengguna IAM atau role yang mengautentikasi permintaan untuk membuat sumber daya. Contoh berikut menggambarkan cara kerjanya:

- Jika Anda menggunakan kredensi pengguna akar akun untuk membuat aturan, akun Anda adalah pemilikEventBridge sumber daya tersebut.
- Jika Anda membuat pengguna di akun Anda dan memberikan izin untuk membuatEventBridge sumber daya untuk pengguna tersebut, pengguna dapat membuatEventBridge sumber daya. Namun, akun Anda, yang memiliki pengguna, memilikiEventBridge sumber daya.
- Jika Anda membuat IAM role di akun Anda dengan izin untuk membuatEventBridge sumber daya, siapa pun yang dapat mengambil peran tersebut dapat membuatEventBridge sumber daya. Akun Anda, yang mana peran tersebut dimiliki, memilikiEventBridge sumber daya.

Mengelola akses ke sumber daya

Kebijakan izin menjelaskan orang yang memiliki akses ke objek. Bagian berikut menjelaskan opsi yang tersedia untuk membuat kebijakan izin.

> **Note**
>
> Bagian ini membahas penggunaan IAM dalam konteks EventBridge. Bagian ini tidak memberikan informasi detail 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 dalam 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. DiEventBridge, Anda dapat menggunakan kebijakan berbasis identitas (kebijakan IAM) dan kebijakan berbasis sumber daya.

Topik

- Kebijakan berbasis identitas (kebijakan IAM)
- Kebijakan berbasis sumber daya (Kebijakan IAM)

Kebijakan berbasis identitas (kebijakan IAM)

Anda dapat melampirkan kebijakan ke identitas IAM. Misalnya, Anda dapat melakukan hal berikut:

- Lampirkan kebijakan izin ke pengguna atau grup dalam akun Anda — Untuk memberikan izin pengguna guna menampilkan aturan dalamCloudWatch konsol Amazon, lampirkan kebijakan izin ke pengguna atau grup tempat pengguna tersebut.
- Lampirkan kebijakan izin untuk peran (memberikan izin lintas akun) – Anda dapat melampirkan kebijakan izin berbasis identitas ke IAM role untuk memberikan izin lintas akun. Misalnya, administrator dalam akun A dapat membuat peran untuk memberikan izin lintas akun ke akun B lain atau layanan AWS sebagai berikut:

  1. Administrator akun A membuat IAM role dan melampirkan kebijakan izin untuk peran yang memberikan izin pada sumber daya di akun A.
  2. Administrator akun A melampirkan kebijakan kepercayaan pada akun identifikasi peran B sebagai prinsipal yang dapat menjalankan peran tersebut.
  3. Administrator Akun B kemudian dapat mendelegasikan izin untuk mengasumsikan peran pada pengguna dalam akun B. Dengan melakukannya, pengguna dalam akun B dapat membuat atau mengakses sumber daya di akun A. Prinsip dalam kebijakan kepercayaan juga dapat menjadi prinsip layanan AWS jika Anda ingin memberikan izin layanan AWS untuk menjalankan peran tersebut.

  Untuk informasi selengkapnya tentang menggunakan IAM untuk mendelegasikan izin, lihat Manajemen Akses dalam Panduan Pengguna IAM.

Anda dapat membuat kebijakan IAM tertentu untuk membatasi panggilan dan sumber daya yang dapat diakses pengguna dalam akun Anda dan kemudian melampirkan kebijakan tersebut ke pengguna. Untuk informasi selengkapnya tentang cara membuat peran IAM dan untuk mengeksplorasi pernyataan kebijakan IAM untukEventBridge, lihatMengelola izin akses keEventBridge sumber daya Amazon Anda.

Kebijakan berbasis sumber daya (Kebijakan IAM)

Ketika aturan berjalanEventBridge, semua target yang terkait dengan aturan dipanggil, yang berarti memanggilAWS Lambda fungsi, menerbitkan ke topik Amazon SNS, atau menyampaikan peristiwa ke aliran Amazon Kinesis. Untuk membuat panggilan API pada sumber daya yang Anda miliki,EventBridge memerlukan izin yang sesuai. Untuk sumber daya Lambda, Amazon SNS, dan Amazon SQS,EventBridge gunakan kebijakan berbasis sumber daya. Untuk aliran Kinesis,EventBridge gunakan peran IAM.

Untuk informasi selengkapnya tentang cara membuat peran IAM dan untuk mengeksplorasi pernyataan kebijakan berbasis sumber dayaEventBridge, lihatMenggunakan kebijakan berbasis sumber daya untuk AmazonEventBridge.

Menentukan elemen kebijakan: tindakan, efek, dan prinsip

Untuk setiapEventBridge sumber daya,EventBridge menentukan serangkaian operasi API. Untuk memberikan izin bagi operasi API ini, EventBridge menentukan serangkaian tindakan yang dapat Anda tentukan dalam kebijakan. Beberapa operasi API memerlukan izin untuk lebih dari satu tindakan untuk melakukan operasi API. Untuk informasi selengkapnya tentang sumber daya dan operasi API, lihat Sumber daya EventBridge dan Referensi izin Amazon EventBridge.

Berikut ini adalah elemen-elemen kebijakan dasar:

- Sumber Daya – Gunakan Amazon Resource Name (ARN) (Amazon Resource Name (ARN)) untuk mengidentifikasi sumber daya yang mengikuti kebijakan tersebut. Untuk informasi selengkapnya, lihat Sumber daya EventBridge.
- Tindakan – Gunakan kata kunci tindakan untuk mengidentifikasi operasi sumber daya yang ingin Anda izinkan atau tolak. Misalnya, izin `events:Describe` memungkinkan pengguna untuk melakukan operasi `Describe`.
- Efek— Tentukan apakah izinkan atau tolak. Jika Anda tidak secara eksplisit memberikan akses ke (izinkan) sumber daya, akses akan ditolak. Anda juga dapat secara eksplisit menolak akses ke sumber daya, yang mungkin Anda lakukan untuk memastikan bahwa pengguna tidak dapat mengaksesnya, meskipun kebijakan yang berbeda memberikan akses.
- Prinsipal – Dalam kebijakan berbasis identitas (kebijakan IAM), pengguna yang kebijakannya terlampir adalah prinsipal implisit. Untuk kebijakan berbasis sumber daya, Anda menentukan pengguna, akun, layanan, atau entitas lain yang ingin Anda terima izinnya (berlaku hanya untuk kebijakan berbasis sumber daya).

Untuk informasi lebih lanjut tentang sintaksis dan deskripsi kebijakan IAM, lihat Referensi Kebijakan IAM JSON dalam Panduan Pengguna IAM.

Untuk informasi tentang tindakanEventBridge API dan sumber daya yang diterapkan, lihatReferensi izin Amazon EventBridge.

Menetapkan ketentuan dalam kebijakan

Ketika Anda memberikan izin, Anda dapat menggunakan bahasa kebijakan akses untuk menentukan syarat ketika kebijakan akan berlaku. Misalnya, Anda mungkin ingin kebijakan diterapkan hanya setelah tanggal tertentu. Untuk informasi selengkapnya tentang menetapkan syarat dalam bahasa kebijakan, lihat Syarat dalam Panduan Pengguna IAM.

Untuk menyatakan syarat, Anda menggunakan kunci syarat. Ada kunciAWS syarat dan kunciEventBridge tertentu yang dapat Anda gunakan sewajarnya. Untuk daftar lengkap kunci AWS, lihat Kunci yang Tersedia untuk Ketentuan dalam Panduan Pengguna IAM. Untuk daftar lengkap kunciEventBridge tertentu, lihatMenggunakan ketentuan kebijakan IAM ketentuan untuk kontrol akses yang sangat baik.

Menggunakan kebijakan berbasis identitas (kebijakan IAM) untuk Amazon EventBridge

Kebijakan berbasis identitas adalah kebijakan izin yang Anda lampirkan ke identitas IAM.

Topik

- AWSkebijakan terkelola untuk EventBridge
- Izin yang diperlukan EventBridge untuk mengakses target menggunakan IAM role
- Contoh kebijakan terkelola pelanggan: Menggunakan penandaan untuk mengontrol akses ke aturan
- EventBridge Pembaruan Amazon untuk kebijakan yangAWS dikelola

AWSkebijakan terkelola untuk EventBridge

AWS menangani banyak kasus penggunaan umum dengan menyediakan kebijakan IAM yang berdiri sendiri, yang dibuat dan dikelola oleh AWS. Kebijakan terkelola, atau yang ditentukan sebelumnya memberikan izin yang diperlukan untuk kasus penggunaan umum sehingga Anda tidak perlu menyelidiki izin yang diperlukan. Untuk informasi lebih lanjut, lihat AWS kebijakan terkelola dalam Panduan Pengguna IAM.

KebijakanAWS terkelola berikut, yang dapat Anda lampirkan ke pengguna di akun Anda, khusus untuk EventBridge:

- AmazonEventBridgeFullAccess- Memberikan akses penuh ke EventBridge, termasuk EventBridge Pipa, EventBridge Skema, dan EventBridge Penjadwal.
- AmazonEventBridgeReadOnlyAccess- Memberikan akses hanya-baca ke EventBridge, termasuk EventBridge Pipa, EventBridge Skema, dan EventBridge Penjadwal.

AmazonEventBridgeFullAccess kebijakan

AmazonEventBridgeFullAccess Kebijakan ini memberikan izin untuk menggunakan semua EventBridge tindakan, serta izin berikut:

- `iam:CreateServiceLinkedRole`— EventBridge memerlukan izin ini untuk membuat peran layanan di akun Anda untuk tujuan API. Izin ini hanya memberikan izin layanan IAM untuk membuat peran dalam akun Anda khusus untuk tujuan API.

- `iam:PassRole`— EventBridge memerlukan izin ini untuk melewati peran EventBridge permintaan untuk meminta target aturan.
- Izin Secrets Manager — EventBridge memerlukan izin ini untuk mengelola rahasia di akun Anda saat Anda memberikan kredensi melalui sumber daya koneksi untuk mengotorisasi Tujuan API.

JSON berikut menunjukkan AmazonEventBridgeFullAccess kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EventBridgeActions",
            "Effect": "Allow",
            "Action": [
                "events:*",
                "schemas:*",
                "scheduler:*",
                "pipes:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "IAMCreateServiceLinkedRoleForApiDestinations",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/
AmazonEventBridgeApiDestinationsServiceRolePolicy",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "apidestinations.events.amazonaws.com"
                }
            }
        },
        {
            "Sid": "IAMCreateServiceLinkedRoleForAmazonEventBridgeSchemas",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/schemas.amazonaws.com/
AWSServiceRoleForSchemas",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "schemas.amazonaws.com"
```

```
                }
            }
        },
        {
            "Sid": "SecretsManagerAccessForApiDestinations",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:CreateSecret",
                "secretsmanager:UpdateSecret",
                "secretsmanager:DeleteSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue"
            ],
            "Resource": "arn:aws:secretsmanager:*:*:secret:events!*"
        },
        {
            "Sid": "IAMPassRoleAccessForEventBridge",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "events.amazonaws.com"
                }
            }
        },
        {
            "Sid": "IAMPassRoleAccessForScheduler",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "scheduler.amazonaws.com"
                }
            }
        },
        {
            "Sid": "IAMPassRoleAccessForPipes",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/*",
            "Condition": {
                "StringLike": {
```

```
                    "iam:PassedToService": "pipes.amazonaws.com"
                }
            }
        }
    ]
}
```

> **Note**
>
> Informasi di bagian ini juga berlaku untuk`CloudWatchEventsFullAccess` kebijakan. Namun, sangat disarankan agar Anda menggunakan Amazon EventBridge alih-alih Amazon CloudWatch Events.

AmazonEventBridgeReadOnlyAccess kebijakan

AmazonEventBridgeReadOnlyAccess Kebijakan ini memberikan izin untuk menggunakan semua EventBridge tindakan baca.

JSON berikut menunjukkan AmazonEventBridgeReadOnlyAccess kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "events:DescribeRule",
                "events:DescribeEventBus",
                "events:DescribeEventSource",
                "events:ListEventBuses",
                "events:ListEventSources",
                "events:ListRuleNamesByTarget",
                "events:ListRules",
                "events:ListTargetsByRule",
                "events:TestEventPattern",
                "events:DescribeArchive",
                "events:ListArchives",
                "events:DescribeReplay",
                "events:ListReplays",
                "events:DescribeConnection",
                "events:ListConnections",
```

```
                "events:DescribeApiDestination",
                "events:ListApiDestinations",
                "events:DescribeEndpoint",
                "events:ListEndpoints",
                "schemas:DescribeCodeBinding",
                "schemas:DescribeDiscoverer",
                "schemas:DescribeRegistry",
                "schemas:DescribeSchema",
                "schemas:ExportSchema",
                "schemas:GetCodeBindingSource",
                "schemas:GetDiscoveredSchema",
                "schemas:GetResourcePolicy",
                "schemas:ListDiscoverers",
                "schemas:ListRegistries",
                "schemas:ListSchemas",
                "schemas:ListSchemaVersions",

                "schemas:ListTagsForResource",
                "schemas:SearchSchemas",
                "scheduler:GetSchedule",
                "scheduler:GetScheduleGroup",
                "scheduler:ListSchedules",
                "scheduler:ListScheduleGroups",
                "scheduler:ListTagsForResource",
                "pipes:DescribePipe",
                "pipes:ListPipes",
                "pipes:ListTagsForResource"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Informasi di bagian ini juga berlaku untuk`CloudWatchEventsReadOnlyAccess` kebijakan. Namun, sangat disarankan agar Anda menggunakan Amazon EventBridge alih-alih Amazon CloudWatch Events.

EventBridge Kebijakan terkelola khusus skema

Skema menetapkan struktur peristiwa yang dikirim ke EventBridge. EventBridge menyediakan skema untuk semua peristiwa yang dihasilkan olehAWS layanan. KebijakanAWS terkelola berikut khusus untuk EventBridge Skema tersedia:

- AmazonEventBridgeSchemasServiceRolePolicy
- AmazonEventBridgeSchemasFullAccess
- AmazonEventBridgeSchemasReadOnlyAccess

EventBridge Kebijakan terkelola khusus penjadwal khusus

Amazon EventBridge Scheduler adalah penjadwal tanpa server yang memungkinkan Anda membuat, menjalankan, dan mengelola tugas dari satu layanan terpusat yang dikelola. Untuk kebijakanAWS terkelola yang khusus untuk EventBridge Penjadwal, lihat kebijakanAWS terkelola untuk EventBridge EventBridge Penjadwal di Panduan Pengguna Penjadwal.

EventBridge Kebijakan terkelola khusus pipa

Amazon EventBridge Pipes menghubungkan sumber peristiwa ke target. Pipa mengurangi kebutuhan akan pengetahuan khusus dan kode integrasi saat mengembangkan arsitektur yang digerakkan oleh acara. Ini membantu memastikan konsistensi di seluruh aplikasi perusahaan Anda. KebijakanAWS terkelola berikut khusus untuk EventBridge Pipa tersedia:

- AmazonEventBridgePipesFullAccess

  Menyediakan akses penuh ke Amazon EventBridge Pipes.

  > **Note**
  >
  > Kebijakan ini menyediakan`iam:PassRole` - EventBridge Pipa memerlukan izin ini untuk meneruskan peran pemanggilan EventBridge untuk membuat, dan memulai pipa.

- AmazonEventBridgePipesReadOnlyAccess

  Menyediakan akses hanya-baca ke Amazon EventBridge Pipes.

- AmazonEventBridgePipesOperatorAccess

  Menyediakan akses read-only dan operator (yaitu, kemampuan untuk menghentikan dan mulai menjalankan Pipes) ke Amazon EventBridge Pipes.

Peran IAM untuk mengirim peristiwa

Untuk relay peristiwa ke target, EventBridge membutuhkan IAM role.

Untuk membuat IAM role untuk mengirim peristiwa ke EventBridge

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Untuk membuat IAM role, ikuti langkah-langkah yang dijelaskan dalam Membuat Peran untuk Mendelegasikan Izin atas AWS Layanan dalam Panduan Pengguna IAM. Saat Anda mengikuti langkah-langkahnya, lakukan hal berikut:
   - Dalam Nama Peran, gunakan nama yang unik dalam akun Anda.
   - Di Pilih Jenis Peran, pilih PeranAWS Layanan, lalu pilih Amazon EventBridge. Tindakan ini memberikan EventBridge izin untuk mengambil peran tersebut.
   - Di Lampirkan Kebijakan, pilih AmazonEventBridgeFullAccess.

Anda juga dapat membuat kebijakan IAM khusus milik Anda sendiri untuk memberikan izin untuk EventBridge tindakan dan sumber daya. Anda dapat melampirkan kebijakan kustom ini ke pengguna IAM atau grup yang memerlukan izin tersebut. Untuk informasi lebih lanjut tentang kebijakan IAM, lihat Gambaran umum Kebijakan IAM dalam Panduan Pengguna IAM. Untuk informasi selengkapnya tentang pengelolaan dan pembuatan kebijakan IAM khusus, lihat Mengelola Kebijakan IAM dalam Panduan Pengguna IAM.

Izin yang diperlukan EventBridge untuk mengakses target menggunakan IAM role

EventBridge target biasanya memerlukan peran IAM yang memberikan izin EventBridge untuk memanggil target. Berikut adalah beberapa contoh untuk berbagaiAWS layanan dan target. Bagi yang lain, gunakan EventBridge konsol untuk membuat Aturan dan buat Peran baru yang akan dibuat dengan kebijakan dengan izin yang memiliki cakupan yang telah dikonfigurasi sebelumnya.

Target Amazon SQS, Amazon SNS, Lambda, CloudWatch Log, dan EventBridge bus tidak menggunakan peran, dan izin yang EventBridge harus diberikan melalui kebijakan sumber daya. Target API Gateway dapat menggunakan kebijakan sumber daya atau peran IAM.

Jika target adalah tujuan API, peran yang Anda tentukan harus menyertakan kebijakan berikut.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
             "Action": [ "events:InvokeApiDestination" ],
            "Resource": [ "arn:aws:events:::api-destination/*" ]
        }
     ]
}
```

Jika target adalah pengaliran Kinesis, peran yang digunakan untuk mengirim data peristiwa ke target tersebut harus menyertakan kebijakan berikut.

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

Jika target adalah Systems Manager run command, dan Anda menentukan satu atau lebih nilai `InstanceIds` untuk perintah, peran yang Anda tentukan harus mencakup kebijakan berikut.

```
{
     "Version": "2012-10-17",
     "Statement": [
           {
                "Action": "ssm:SendCommand",
                "Effect": "Allow",
                "Resource": [
                     "arn:aws:ec2:region:accountId:instance/instanceIds",
                     "arn:aws:ssm:region:*:document/documentName"
                ]
           }
     ]
}
```

Jika target adalah Systems Manager run command, dan Anda menentukan satu atau lebih tag untuk perintah, peran yang Anda tentukan harus menyertakan kebijakan berikut.

```
{
     "Version": "2012-10-17",
     "Statement": [
           {
                "Action": "ssm:SendCommand",
                "Effect": "Allow",
                "Resource": [
                     "arn:aws:ec2:region:accountId:instance/*"
                ],
                "Condition": {
                    "StringEquals": {
                        "ec2:ResourceTag/*": [
                            "[[tagValues]]"
                        ]
                    }
                }
           },
           {
                "Action": "ssm:SendCommand",
                "Effect": "Allow",
                "Resource": [
                     "arn:aws:ssm:region:*:document/documentName"
                ]
           }
     ]
}
```

Jika target adalah AWS Step Functions mesin stata, peran yang Anda tentukan harus mencakup kebijakan berikut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
             "Action": [ "states:StartExecution" ],
            "Resource": [ "arn:aws:states:*:*:stateMachine:*" ]
        }
     ]
```

```
}
```

Jika target adalah tugas Amazon ECS, peran yang Anda tentukan harus menyertakan kebijakan berikut.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "ecs:RunTask"
        ],
        "Resource": [
            "arn:aws:ecs:*:account-id:task-definition/task-definition-name"
        ],
        "Condition": {
            "ArnLike": {
                "ecs:cluster": "arn:aws:ecs:*:account-id:cluster/cluster-name"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action":"iam:PassRole",
        "Resource": [
            "*"
        ],
        "Condition": {
            "StringLike": {
                "iam:PassedToService": "ecs-tasks.amazonaws.com"
            }
        }
    }]
}
```

Kebijakan berikut memungkinkan built-in target masuk EventBridge untuk melakukan tindakan Amazon EC2 atas nama Anda. Anda harus menggunakan AWS Management Console untuk membuat aturan dengan built-in target.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "TargetInvocationAccess",
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:RebootInstances",
                "ec2:StopInstances",
                "ec2:TerminateInstances",
                "ec2:CreateSnapshot"
            ],
            "Resource": "*"
        }
    ]
}
```

Kebijakan berikut memungkinkan EventBridge relay peristiwa ke aliran Kinesis di akun Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "KinesisAccess",
            "Effect": "Allow",
            "Action": [
                "kinesis:PutRecord"
            ],
            "Resource": "*"
        }
    ]
}
```

Contoh kebijakan terkelola pelanggan: Menggunakan penandaan untuk mengontrol akses ke aturan

Contoh berikut menunjukkan kebijakan pengguna yang memberikan izin untuk EventBridge tindakan. Kebijakan ini bekerja saat Anda menggunakan EventBridge API,AWS SDK, atauAWS CLI.

Anda dapat memberi pengguna akses ke EventBridge aturan tertentu serta mencegah mereka mengakses aturan lain. Untuk melakukannya, Anda tandai kedua set aturan dan kemudian gunakan kebijakan IAM yang merujuk ke tag tersebut. Untuk informasi selengkapnya tentang penandaan EventBridge sumber daya, lihat EventBridge Tag Amazon.

Anda dapat memberikan kebijakan IAM untuk pengguna untuk mengizinkan akses ke hanya aturan dengan tanda tertentu. Anda memilih aturan untuk memberikan akses ke dengan memberi menandai mereka dengan tanda tertentu. Sebagai contoh, kebijakan berikut memberikan akses pengguna ke aturan dengan nilai `Prod` untuk kunci tanda`Stack`.

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "events:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Stack": "Prod"
                }
            }
        }
    ]
}
```

Untuk informasi selengkapnya tentang menggunakan pernyataan kebijakan IAM, lihat Mengontrol Akses Menggunakan Kebijakan di Panduan Pengguna IAM.

EventBridge Pembaruan Amazon untuk kebijakan yangAWS dikelola

Lihat detail tentang pembaruan ke kebijakanAWS terkelola EventBridge karena layanan ini mulai melacak perubahan ini. Untuk pemberitahuan otomatis tentang perubahan pada halaman ini, berlangganan umpan RSS di halaman Riwayat EventBridge dokumen.

Perubahan	Deskripsi	Tanggal
AmazonEventBridgePipesFullAccess- Kebijakan baru ditambahkan	EventBridge menambahkan kebijakan terkelola untuk izin penuh untuk menggunakan EventBridge Pipa.	1 Desember 2022
AmazonEventBridgePipesReadOnlyAccess- Kebijakan baru ditambahkan	EventBridge menambahkan kebijakan terkelola untuk	1 Desember 2022

Perubahan	Deskripsi	Tanggal
	izin untuk melihat sumber informasi EventBridge Pipa.	
AmazonEventBridgePipesOperatorAccess- Kebijakan baru ditambahkan	EventBridge menambahkan kebijakan terkelola untuk izin untuk melihat informasi EventBridge Pipa, serta mulai dan berhenti menjalankan pipa.	1 Desember 2022
AmazonEventBridgeFullAccess — Permbaruan ke kebijakan yang sudah ada	EventBridge memperbarui kebijakan untuk menyertakan izin yang diperlukan untuk menggunakan fitur EventBridge Pipes.	1 Desember 2022
AmazonEventBridgeReadOnlyAccess — Permbaruan ke kebijakan yang sudah ada	EventBridge menambahkan izin yang diperlukan untuk melihat EventBridge pipa sumber informasi. Tindakan berikut ditambahkan: • `pipes:DescribePipe` • `pipes:ListPipes` • `pipes:ListTagsForResource`	1 Desember 2022
CloudWatchEventsReadOnlyAccess — Permbaruan ke kebijakan yang sudah ada	Diperbarui untuk mencocokkan AmazonEventBridgeReadOnlyAccess.	1 Desember 2022
CloudWatchEventsFullAccess — Permbaruan ke kebijakan yang sudah ada	Diperbarui untuk mencocokkan AmazonEventBridgeFullAccess.	1 Desember 2022

Perubahan	Deskripsi	Tanggal
AmazonEventBridgeFullAccess — Permbaruan ke kebijakan yang sudah ada	EventBridge memperbarui kebijakan untuk menyertakan izin yang diperlukan untuk menggunakan skema dan fitur penjadwal. Izin berikut ditambahkan: • EventBridge Tindakan Registri Skema • EventBridge Tindakan penjadwal • `iam:CreateServiceLinkedRole` izin untuk EventBridge Schema Registry • `iam:PassRole` izin untuk EventBridge Scheduler	10 November 2022

Perubahan	Deskripsi	Tanggal
AmazonEventBridgeReadOnlyAccess — Pembaruan ke kebijakan yang sudah ada	EventBridge izin tambahan yang diperlukan untuk melihat skema dan sumber informasi scheduler. Tindakan berikut ditambahkan: • `schemas:DescribeCodeBinding` • `schemas:DescribeDiscoverer` • `schemas:DescribeRegistry` • `schemas:DescribeSchema` • `schemas:ExportSchema` • `schemas:GetCodeBindingSource` • `schemas:GetDiscoveredSchema` • `schemas:GetResourcePolicy` • `schemas:ListDiscoverers` • `schemas:ListRegistries` • `schemas:ListSchemas` • `schemas:ListSchemaVersions` • `schemas:ListTagsForResource`	10 November 2022

Perubahan	Deskripsi	Tanggal
	• `schemas:SearchSchemas` • `scheduler:GetSchedule` • `scheduler:GetScheduleGroup` • `scheduler:ListSchedules` • `scheduler:ListScheduleGroups` • `scheduler:ListTagsForResource`	
AmazonEventBridgeReadOnlyAccess — Pembaruan ke kebijakan yang sudah ada	EventBridge izin tambahan yang diperlukan untuk melihat informasi endpoint. Tindakan berikut ditambahkan: • `events:ListEndpoints` • `events:DescribeEndpoint`	7 April 2022

Perubahan	Deskripsi	Tanggal
AmazonEventBridgeReadOnlyAccess — Pembaruan ke kebijakan yang sudah ada	EventBridge izin tambahan yang diperlukan untuk melihat koneksi dan informasi tujuan API. Tindakan berikut ditambahkan: • `events:DescribeConnection` • `events:ListConnections` • `events:DescribeApiDestination` • `events:ListApiDestinations`	4 Maret 2021

Perubahan	Deskripsi	Tanggal
AmazonEventBridgeFullAccess — Permbaruan ke kebijakan yang sudah ada	EventBridge memperbarui kebijakan untuk menyertakan `aniam:CreateServiceLinkedRole` dan AWS Secrets Manager izin yang diperlukan untuk menggunakan tujuan API. Tindakan berikut ditambahkan: • `secretsmanager:CreateSecret` • `secretsmanager:UpdateSecret` • `secretsmanager:DeleteSecret` • `secretsmanager:GetSecretValue` • `secretsmanager:PutSecretValue`	4 Maret 2021
EventBridge mulai melacak perubahan	EventBridge mulai melacak perubahan untuk kebijakan AWS terkelola.	4 Maret 2021

Menggunakan kebijakan berbasis sumber daya untuk AmazonEventBridge

Saat aturan berjalanEventBridge, semua target terkait dengan aturan dipanggil. Aturan dapat memanggil fungsi AWS Lambda, memublikasikan ke topik Amazon SNS, atau me-relay peristiwa tersebut ke aliran Kinesis. Untuk membuat panggilan API terhadap sumber daya yang Anda miliki,EventBridge perlu izin yang sesuai. Untuk sumber daya Lambda, Amazon SNS, Amazon SQS, dan AmazonCloudWatch Logs,EventBridge menggunakan kebijakan berbasis sumber daya. Untuk aliran Kinesis,EventBridge gunakan kebijakan berbasis identitas.

Anda menggunakan AWS CLI untuk menambahkan izin ke target Anda. Untuk informasi tentang cara memasang dan mengonfigurasi AWS CLI, lihat Menyiapkan Pengaturan dengan AWS Command Line Interface dalam AWS Command Line Interface Panduan Pengguna.

Topik

- Izin Amazon API Gateway
- CloudWatchIzin log
- Izin AWS Lambda
- Izin Amazon SNS
- Izin Amazon SQS
- EventBridgePipa spesifik

Izin Amazon API Gateway

Untuk meminta titik akhir Amazon API Gateway Anda dengan menggunakanEventBridge aturan, tambahkan izin berikut untuk kebijakan titik akhir API Gateway Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
       {
           "Effect": "Allow",
           "Principal": {
               "Service": "events.amazonaws.com"
           },
           "Action": "execute-api:Invoke",
           "Condition": {
               "ArnEquals": {
                   "aws:SourceArn": "arn:aws:events:region:account-id:rule/rule-name"
```

```
                }
            },
            "Resource": [
                "execute-api:/stage/GET/api"
            ]
        }
    ]
}
```

CloudWatchIzin log

SaatCloudWatch log menjadi target aturan,EventBridge buat pengaliran log, danCloudWatch log menyimpan teks dari peristiwa sebagai entri log. EventBridgeUntuk mengizinkan pengaliran log dan mencatat peristiwa,CloudWatch log harus menyertakan kebijakan berbasis sumber daya yang memungkinkanEventBridge untuk menulis keCloudWatch log.

Jika Anda menggunakanAWS Management Console untuk menambahkanCloudWatch log sebagai target aturan, kebijakan berbasis sumber daya dibuat secara otomatis. Jika Anda menggunakan AWS CLI untuk menambahkan target, dan kebijakan belum ada, Anda harus membuatnya.

Contoh berikut memungkinkanEventBridge untuk menulis ke semua grup log yang memiliki nama yang dimulai dengan`/aws/events/`. Jika Anda menggunakan kebijakan penamaan yang berbeda untuk jenis log ini, sesuaikan contoh yang sesuai.

```
{
    "Statement": [
        {
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Effect": "Allow",
            "Principal": {
                "Service": ["events.amazonaws.com", "delivery.logs.amazonaws.com"]
            },
            "Resource": "arn:aws:logs:region:account:log-group:/aws/events/*:*",
            "Sid": "TrustEventsToStoreLogEvent"
        }
    ],
    "Version": "2012-10-17"
}
```

Untuk informasi selengkapnya, lihat PutResourcePolicydi panduan ReferensiCloudWatch Logs API.

Izin AWS Lambda

Untuk mengaktifkanAWS Lambda fungsi Anda dengan menggunakanEventBridge aturan, tambahkan izin berikut untuk kebijakan fungsi Lambda Anda.

```
{
  "Effect": "Allow",
  "Action": "lambda:InvokeFunction",
  "Resource": "arn:aws:lambda:region:account-id:function:function-name",
  "Principal": {
    "Service": "events.amazonaws.com"
  },
  "Condition": {
    "ArnLike": {
      "AWS:SourceArn": "arn:aws:events:region:account-id:rule/rule-name"
    }
  },
  "Sid": "InvokeLambdaFunction"
}
```

Untuk menambahkan izin di atas yang mengaktifkanEventBridge untuk memanggil fungsi LambdaAWS CLI

- Pada jendela perintah, masukkan perintah berikut.

```
aws lambda add-permission --statement-id "InvokeLambdaFunction" \
--action "lambda:InvokeFunction" \
--principal "events.amazonaws.com" \
--function-name "arn:aws:lambda:region:account-id:function:function-name" \
--source-arn "arn:aws:events:region:account-id:rule/rule-name"
```

Untuk informasi selengkapnya tentang pengaturan izin yang memungkinkanEventBridge untuk menjalankan fungsi Lambda, lihat AddPermissiondan Menggunakan Lambda dengan Acara Terjadwal di PanduanAWS Lambda Pengembang.

Izin Amazon SNS

Untuk memungkinkanEventBridge untuk mempublikasikan ke topik Amazon SNS, gunakan`aws sns get-topic-attributes` dan`aws sns set-topic-attributes` perintah.

> **Note**
>
> Anda tidak dapat menggunakan`Condition` blok dalam kebijakan topik Amazon SNS untukEventBridge.

Untuk menambahkan izin yang memungkinkanEventBridge untuk memublikasikan topik SNS

1. Untuk mencantumkan atribut topik SNS, gunakan perintah berikut.

```
aws sns get-topic-attributes --topic-arn "arn:aws:sns:region:account-id:topic-name"
```

   Contoh berikut menunjukkan hasil topik SNS baru.

```
{
    "Attributes": {
        "SubscriptionsConfirmed": "0",
        "DisplayName": "",
        "SubscriptionsDeleted": "0",
        "EffectiveDeliveryPolicy": "{\"http\":{\"defaultHealthyRetryPolicy\":
{\"minDelayTarget\":20,\"maxDelayTarget\":20,\"numRetries\":3,\"numMaxDelayRetries
\":0,\"numNoDelayRetries\":0,\"numMinDelayRetries\":0,\"backoffFunction\":\"linear
\"},\"disableSubscriptionOverrides\":false}}",
        "Owner": "account-id",
        "Policy": "{\"Version\":\"2012-10-17\",\"Id\":\"__default_policy_ID\",
\"Statement\":[{\"Sid\":\"__default_statement_ID\",\"Effect\":\"Allow\",\"Principal
\":{\"AWS\":\"*\"},\"Action\":[\"SNS:GetTopicAttributes\",\"SNS:SetTopicAttributes
\",\"SNS:AddPermission\",\"SNS:RemovePermission\",\"SNS:DeleteTopic\",
\"SNS:Subscribe\",\"SNS:ListSubscriptionsByTopic\",\"SNS:Publish\"],\"Resource
\":\"arn:aws:sns:region:account-id:topic-name\",\"Condition\":{\"StringEquals\":
{\"AWS:SourceOwner\":\"account-id\"}}}]}",
        "TopicArn": "arn:aws:sns:region:account-id:topic-name",
        "SubscriptionsPending": "0"
    }
}
```

2. Gunakan Konverter JSON ke string untuk mengonversi pernyataan berikut ke string.

```
{
  "Sid": "PublishEventsToMyTopic",
  "Effect": "Allow",
```

```
  "Principal": {
    "Service": "events.amazonaws.com"
  },
  "Action": "sns:Publish",
  "Resource": "arn:aws:sns:region:account-id:topic-name"
}
```

Setelah Anda mengonversi pernyataan ke string, hasilnya seperti contoh berikut.

```
{\"Sid\":\"PublishEventsToMyTopic\",\"Effect\":\"Allow\",\"Principal\":
{\"Service\":\"events.amazonaws.com\"},\"Action\":\"sns:Publish\",\"Resource\":
\"arn:aws:sns:region:account-id:topic-name\"}
```

3. Tambahkan string yang Anda buat pada langkah sebelumnya ke `"Statement"`kumpulan di dalam `"Policy"` atribut.
4. Gunakan `aws sns set-topic-attributes` perintah untuk menetapkan kebijakan baru.

```
aws sns set-topic-attributes --topic-arn "arn:aws:sns:region:account-id:topic-name"
 \
--attribute-name Policy \
--attribute-value "{\"Version\":\"2012-10-17\",\"Id\":\"__default_policy_ID\",
\"Statement\":[{\"Sid\":\"__default_statement_ID\",\"Effect\":\"Allow\",\"Principal
\":{\"AWS\":\"*\"},\"Action\":[\"SNS:GetTopicAttributes\",\"SNS:SetTopicAttributes
\",\"SNS:AddPermission\",\"SNS:RemovePermission\",\"SNS:DeleteTopic\",
\"SNS:Subscribe\",\"SNS:ListSubscriptionsByTopic\",\"SNS:Publish\"],\"Resource
\":\"arn:aws:sns:region:account-id:topic-name\",\"Condition\":{\"StringEquals
\":{\"AWS:SourceOwner\":\"account-id\"}}}, {\"Sid\":\"PublishEventsToMyTopic\",
\"Effect\":\"Allow\",\"Principal\":{\"Service\":\"events.amazonaws.com\"},\"Action
\":\"sns:Publish\",\"Resource\":\"arn:aws:sns:region:account-id:topic-name\"}]}"
```

Untuk informasi selengkapnya, lihat SetTopicAttributestindakan di Amazon Simple Notification Service API Reference.

Izin Amazon SQS

Untuk mengizinkanEventBridge aturan menjalankan antrean Amazon SQS, gunakan`aws sqs get-queue-attributes` dan`aws sqs set-queue-attributes` perintah.

Jika kebijakan untuk antrean SQS kosong, Anda harus terlebih dahulu membuat kebijakan dan kemudian Anda dapat menambahkan pernyataan izin untuk itu. Antrean SQS baru memiliki kebijakan kosong.

Jika antrean SQS sudah memiliki kebijakan, Anda perlu menyalin kebijakan asli dan menggabungkannya dengan pernyataan baru untuk menambahkan pernyataan izin untuk itu.

Untuk menambahkan izin yang mengaktifkanEventBridge aturan untuk memanggil antrean SQS

1. Untuk mencantumkan atribut antrean SQS. Pada jendela perintah, masukkan perintah berikut.

```
aws sqs get-queue-attributes \
--queue-url https://sqs.region.amazonaws.com/account-id/queue-name \
--attribute-names Policy
```

2. Tambahkan pernyataan berikut.

```
{
      "Sid": "AWSEvents_custom-eventbus-ack-sqs-rule_dlq_sqs-rule-target",
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Action": "sqs:SendMessage",
      "Resource": "arn:aws:sqs:region:account-id:queue-name",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": "arn:aws:events:region:account-id:rule/bus-name/rule-
name"
        }
      }
    }
```

3. Gunakan Konverter JSON ke string untuk mengonversi pernyataan sebelumnya menjadi string. Setelah Anda mengonversi kebijakan ke string, hasilnya terlihat seperti berikut.

```
{\"Sid\": \"EventsToMyQueue\", \"Effect\": \"Allow\", \"Principal\": {\"Service
\": \"events.amazonaws.com\"}, \"Action\": \"sqs:SendMessage\", \"Resource\":
 \"arn:aws:sqs:region:account-id:queue-name\", \"Condition\": {\"ArnEquals\":
 {\"aws:SourceArn\": \"arn:aws:events:region:account-id:rule/rule-name\"}}
```

4. Buat file bernama `set-queue-attributes.json` dengan konten berikut.

```
{
    "Policy": "{\"Version\":\"2012-10-17\",\"Id\":\"arn:aws:sqs:region:account-
id:queue-name/SQSDefaultPolicy\",\"Statement\":[{\"Sid\": \"EventsToMyQueue\",
 \"Effect\": \"Allow\", \"Principal\": {\"Service\": \"events.amazonaws.com\"},
 \"Action\": \"sqs:SendMessage\", \"Resource\": \"arn:aws:sqs:region:account-
id:queue-name\", \"Condition\": {\"ArnEquals\": {\"aws:SourceArn\":
 \"arn:aws:events:region:account-id:rule/rule-name\"}}}]}"
}
```

5. Menetapkan atribut kebijakan dengan menggunakan file `set-queue-attributes.json` yang baru saja dibuat sebagai input, seperti yang ditunjukkan dalam perintah berikut.

```
aws sqs set-queue-attributes \
--queue-url https://sqs.region.amazonaws.com/account-id/queue-name \
--attributes file://set-queue-attributes.json
```

Untuk informasi lebih lanjut, lihat Amazon SQS Policy Examples di Panduan Developer Amazon Simple Queue Service.

EventBridgePipa spesifik

EventBridgePipa tidak mendukung kebijakan berbasis sumber daya dan tidak memiliki API yang mendukung kondisi kebijakan berbasis sumber daya.

Pencegahan Deputi Bingung Lintas Layanan

Masalah deputi yang membingungkan adalah masalah keamanan di mana entitas yang tidak memiliki izin untuk melakukan tindakan dapat memaksa entitas yang lebih istimewa untuk melakukan tindakan. DiAWS, peniruan lintas layanan dapat mengakibatkan masalah wakil yang membingungkan. Peniruan identitas lintas layanan dapat terjadi ketika satu layanan (layanan panggilan) memanggil layanan lain (disebut layanan). Layanan panggilan dapat dimanipulasi untuk menggunakan izinnya untuk bertindak atas sumber daya pelanggan lain dengan cara yang seharusnya tidak memiliki izin untuk mengakses. Untuk mencegah hal ini,AWSmenyediakan alat yang membantu Anda melindungi data Anda untuk semua layanan dengan prinsipal layanan yang telah diberikan akses ke sumber daya di akun Anda.

Kami merekomendasikan menggunakan`aws:SourceArn`dan`aws:SourceAccount`kunci konteks kondisi global dalam kebijakan sumber daya untuk membatasi izin yang dimiliki

AmazonEventBridgememberikan layanan lain ke sumber daya. Gunakan`aws:SourceArn`jika Anda hanya ingin satu sumber daya dikaitkan dengan akses lintas layanan. Gunakan`aws:SourceAccount`jika Anda ingin mengizinkan sumber daya apa pun di akun itu dikaitkan dengan penggunaan lintas layanan.

Cara paling efektif untuk melindungi dari masalah wakil yang membingungkan adalah dengan menggunakan`aws:SourceArn`kunci konteks kondisi global dengan ARN penuh sumber daya. Jika Anda tidak mengetahui ARN lengkap sumber daya atau jika Anda menentukan beberapa sumber daya, gunakan`aws:SourceArn`kunci kondisi konteks global dengan karakter wildcard (*) untuk bagian ARN yang tidak diketahui. Sebagai contoh, `arn:aws:`*`servicename`*`:*:`*`123456789012`*`:*`.

Jika`aws:SourceArn`value tidak berisi ID akun, seperti bucket ARN Amazon S3, Anda harus menggunakan kedua kunci konteks kondisi global untuk membatasi izin.

Bus peristiwa

UntukEventBridgetarget aturan bus acara, nilai`aws:SourceArn`harus menjadi aturan ARN.

Contoh berikut menunjukkan bagaimana Anda dapat menggunakan`aws:SourceArn`dan`aws:SourceAccount`kunci konteks kondisi global diEventBridgeuntuk mencegah masalah wakil yang membingungkan. Contoh ini untuk digunakan dalam kebijakan kepercayaan peran, untuk peran yang digunakan olehEventBridgeaturan.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "events.amazonaws.com"
    },
    "Action": "sts:AssumeRole"
    ],
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:events:*:123456789012:rule/myRule"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
```

```
  }
}
```

EventBridgePipa

UntukEventBridgePipa, nilai`aws:SourceArn`harus pipa ARN.

Contoh berikut menunjukkan bagaimana Anda dapat menggunakan`aws:SourceArn`dan`aws:SourceAccount`kunci konteks kondisi global diEventBridgeuntuk mencegah masalah wakil yang membingungkan. Contoh ini untuk digunakan dalam kebijakan kepercayaan peran, untuk peran yang digunakan olehEventBridgePipa.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "events.amazonaws.com"
    },
    "Action": "sts:AssumeRole"
    ],
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:pipe:*:123456789012::pipe/example"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

Kebijakan berbasis sumber daya untuk skema Amazon EventBridge

Registri skema EventBridge mendukung kebijakan berbasis sumber daya. Suatu kebijakan berbasis sumber dayaadalah kebijakan yang melekat pada sumber daya, bukan identitas IAM. Misalnya, di Amazon Simple Storage Service (Amazon S3), kebijakan sumber daya melekat pada bucket Amazon S3.

Untuk informasi selengkapnya tentang Skema EventBridge dan kebijakan berbasis sumber daya, lihat hal berikut.

- Referensi REST API Skema Amazon EventBridge
- Kebijakan Berbasis Identitas dan Kebijakan Berbasis Sumber Daya dalam Panduan Pengguna IAM

API yang didukung untuk kebijakan berbasis sumber daya

Anda dapat menggunakan API berikut dengan kebijakan berbasis sumber daya untuk registri skema EventBridge.

- `DescribeRegistry`
- `UpdateRegistry`
- `DeleteRegistry`
- `ListSchemas`
- `SearchSchemas`
- `DescribeSchema`
- `CreateSchema`
- `DeleteSchema`
- `UpdateSchema`
- `ListSchemaVersions`
- `DeleteSchemaVersion`
- `DescribeCodeBinding`
- `GetCodeBindingSource`
- `PutCodeBinding`

Contoh kebijakan yang memberikan semua tindakan yang didukung ke AWS akun

Untuk registri skema EventBridge, Anda harus selalu melampirkan kebijakan berbasis sumber daya ke registri. Untuk memberikan akses ke skema, Anda menentukan ARN skema dan ARN registri dalam kebijakan.

Untuk memberikan akses pengguna ke semua API yang tersedia untuk Skema EventBridge, gunakan kebijakan yang mirip dengan berikut ini, yang menggantikan `"Principal"` dengan ID akun dari akun yang ingin Anda berikan akses.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Test",
            "Effect": "Allow",
            "Action": [
                "schemas:*"
            ],
            "Principal": {
                "AWS": [
                    "109876543210"
                ]
            },
            "Resource": [
                "arn:aws:schemas:us-east-1:012345678901:registry/default",
                "arn:aws:schemas:us-east-1:012345678901:schema/default*"
            ]
        }
    ]
}
```

Contoh kebijakan yang memberikan tindakan hanya-baca ke AWS akun

Contoh berikut memberikan akses ke akun hanya untuk API hanya-baca untuk skema EventBridge.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Test",
            "Effect": "Allow",
```

```
            "Action": [
                "schemas:DescribeRegistry",
                "schemas:ListSchemas",
                "schemas:SearchSchemas",
                "schemas:DescribeSchema",
                "schemas:ListSchemaVersions",
                "schemas:DescribeCodeBinding",
                "schemas:GetCodeBindingSource"
            ],
            "Principal": {
                "AWS": [
                    "109876543210"
                ]
            },
            "Resource": [
                "arn:aws:schemas:us-east-1:012345678901:registry/default",
                "arn:aws:schemas:us-east-1:012345678901:schema/default*"
            ]
        }
    ]
}
```

Contoh kebijakan yang memberikan semua tindakan untuk organisasi

Anda dapat menggunakan kebijakan berbasis sumber daya dengan registri skema EventBridge untuk memberikan akses ke organisasi. Untuk informasi selengkapnya, lihat AWS Organizations Panduan Pengguna . Contoh berikut memberikan ID `o-a1b2c3d4e5` akses ke registri skema kepada organisasi.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Test",
            "Effect": "Allow",
            "Action": [
                "schemas:*"
            ],
            "Principal": "*",
            "Resource": [
                "arn:aws:schemas:us-east-1:012345678901:registry/default",
                "arn:aws:schemas:us-east-1:012345678901:schema/default*"
            ],
```

```
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": [
                        "o-a1b2c3d4e5"
                    ]
                }
            }
        }
    ]
}
```

Referensi izin Amazon EventBridge

Untuk menentukan tindakan dalam kebijakan EventBridge, gunakan`events:` prefiks diikuti dengan nama operasi API, seperti yang ditunjukkan dalam contoh berikut.

```
"Action": "events:PutRule"
```

Untuk menetapkan beberapa tindakan dalam satu pernyataan, pisahkan dengan koma seperti berikut.

```
"Action": ["events:action1", "events:action2"]
```

Anda juga bisa menggunakan tanda bintang untuk menentukan beberapa tindakan. Misalnya, Anda dapat menentukan semua tindakan yang namanya dimulai dengan kata `"Put"` sebagai berikut.

```
"Action": "events:Put*"
```

Untuk menentukan semua tindakan API EventBridge, gunakan * tanda bintang sebagai berikut.

```
"Action": "events:*"
```

Tabel berikut mencantumkan operasi API EventBridge dan tindakan yang sesuai yang dapat Anda tentukan dalam kebijakan IAM.

Operasi API EventBridge	Izin yang diperlukan	Deskripsi
DeleteRule	`events:DeleteRule`	Diperlukan untuk menghapus aturan.
DescribeEventBus	`events:DescribeEventBus`	Diperlukan untuk mencantumkan akun yang diperbolehkan untuk menulis peristiwa untuk bus peristiwa akun saat ini.
DescribeRule	`events:DescribeRule`	Wajib mencantumkan detail aturan.

Operasi API EventBridge	Izin yang diperlukan	Deskripsi
DisableRule	`events:DisableRule`	Wajib menonaktifkan aturan.
EnableRule	`events:EnableRule`	Diperlukan untuk mengaktif kan aturan.
ListRuleNamesByTarget	`events:ListRuleNam esByTarget`	Wajib mencantumkan aturan terkait target.
ListRules	`events:ListRules`	Diperlukan untuk mencantum kan semua aturan di akun Anda.
ListTagsForResource	`events:ListTagsFor Resource`	Diperlukan untuk mencantum kan semua tag yang terkait dengan sumber daya EventBridge. Saat ini, hanya aturan yang bisa diberi tag.
listTargetsByRule	`events:ListTargets ByRule`	Wajib mencantumkan semua target yang terkait dengan aturan.
PutEvents	`events:PutEvents`	Diperlukan untuk menambahk an peristiwa khusus yang dapat dicocokkan dengan aturan.
PutPermission	`events:PutPermission`	Diperlukan untuk memberika n izin akun lain untuk menulis peristiwa ke bus peristiwa default akun ini.
PutRule	`events:PutRule`	Diperlukan untuk membuat atau memperbarui aturan.
PutTargets	`events:PutTargets`	Diperlukan untuk menambahk an target ke aturan.

Operasi API EventBridge	Izin yang diperlukan	Deskripsi
RemovePermission	`events:RemovePermission`	Diperlukan untuk mencabut izin akun lain untuk menulis peristiwa ke bus peristiwa default akun ini.
RemoveTargets	`events:RemoveTargets`	Diperlukan untuk menghapus target dari aturan.
TestEventPattern	`events:TestEventPattern`	Diperlukan untuk menguji pola peristiwa terhadap peristiwa tertentu.

Menggunakan ketentuan kebijakan IAM ketentuan untuk kontrol akses yang sangat baik

Untuk memberikan izin, Anda menggunakan bahasa kebijakan IAM dalam pernyataan kebijakan untuk menetapkan ketentuan ketika kebijakan akan berlaku. Misalnya, Anda dapat memiliki kebijakan yang diterapkan hanya setelah tanggal tertentu.

Sebuah syarat dalam kebijakan terdiri dari pasangan nilai kunci. Kunci syarat tidak memedulikan huruf besar atau kecil.

Jika Anda menetapkan beberapa syarat atau kunci dalam satu ketentuan, semua ketentuan dan kunci harus dipenuhi EventBridge untuk memberikan izin. Jika Anda menetapkan satu syarat dengan beberapa nilai untuk satu kunci, EventBridge berikan izin jika salah satu nilai terpenuhi.

Anda juga dapat menggunakan placeholder atau variabel kebijakan saat menetapkan ketentuan. Untuk informasi selengkapnya, lihat Variabel Kebijakan dalam Panduan Pengguna IAM. Untuk informasi selengkapnya tentang menetapkan ketentuan dalam bahasa kebijakan IAM, lihat Syarat dalam Panduan Pengguna IAM.

Secara default, pengguna dan peran IAM tidak dapat mengaksesperistiwa di akun Anda. Untuk mengakses peristiwa, pengguna harus berwenang untuk `PutRule`etindakan API. Jika pengguna atau peran IAM diotorisasi untuk tindakan `events:PutRule`, mereka dapat membuat aturan yang cocok dengan peristiwa tertentu. Namun, agar aturan dapat digunakan, pengguna juga harus memiliki izin untuk`events:PutTargets` tindakan karena, jika Anda ingin aturan melakukan lebih dari publikasi CloudWatch metrik, Anda juga harus menambahkan target ke aturan.

Anda dapat memberikan syarat dalam pernyataan kebijakan peran atau pengguna IAM yang memungkinkan pengguna atau peran untuk membuat aturan yang hanya cocok dengan serangkaian sumber dan jenis peristiwa tertentu. Untuk memberikan akses ke sumber dan jenis peristiwa tertentu, gunakan `events:source` dan `events:detail-type` kunci syarat.

Demikian pula, Anda dapat memberikan syarat dalam pernyataan kebijakan pengguna atau peran IAM yang memungkinkan pengguna atau peran untuk membuat aturan yang hanya cocok dengan sumber daya tertentu di akun Anda. Untuk memberikan akses ke sumber daya tertentu, gunakan `events:TargetArn`kunci syarat.

Contoh berikut adalah kebijakan yang memungkinkan pengguna untuk mengakses semua peristiwa kecuali peristiwa Amazon EC2 dalam EventBridge menggunakan pernyataan tolak pada Tindakan`PutRule` API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyPutRuleForAllEC2Events",
            "Effect": "Deny",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:source": "aws.ec2"
                }
            }
        }
    ]
}
```

EventBridge Kunci syarat

Tabel berikut menunjukkan kunci syarat dan pasangan kunci dan nilai yang dapat Anda gunakan dalam kebijakan EventBridge.

Kunci syarat	Pasangan nilai kunci	Jenis evaluasi
aws:SourceAccount	Akun di mana aturan yang ditetapkan oleh `aws:SourceArn` ada.	Account Id, Null
aws:SourceArn	ARN aturan yang mengirimkan acara tersebut.	ARN, Null
events:creatorAccount	`"events:creatorAccount":"` *`creatorAccount`* `"` Untuk *`creatorAccount`* , gunakan ID akun untuk akun yang membuat aturan. Gunakan syarat ini untuk mengesahkan panggilan API pada aturan dari akun tertentu.	creatorAccount, Null

Kunci syarat	Pasangan nilai kunci	Jenis evaluasi
events:detail-type	`"events:detail-type":" detail-type "` Di mana *tipe detail* adalah string literal untuk tipe detailbidang peristiwa seperti `"AWS API Call via CloudTrail"` dan `"EC2 Instance State-change Notification"` .	Detail Type, Null
peristiwa: detail. eventTypeCode	`"events:detail.eventTypeCode":" eventTypeCode "` Untuk *eventTypeCode* , gunakan string literal untuk detailnya. eventTypeCodebidang acara, seperti`"AWS_ABUSE_DOS_REPORT"` .	eventTypeCode, Null
Event: detail.service	`"events:detail.service":" service"` Untuk *layanan*, gunakan string literal untuk detail.servicebidang peristiwa, seperti `"ABUSE"`.	service, Null

Kunci syarat	Pasangan nilai kunci	Jenis evaluasi
events: detail.userIdentity.principalId	`"events:detail.userIdentity.principalId":" principal-id "` Untuk *id-utama*, gunakan string literal untuk bidang Detail.userIdentity.PrincipalID peristiwa dengan tipe detail `"AWS API Call via CloudTrail"` seperti `"AROAIDPPEZS35WEXAMPLE:AssumedRoleSessionName."` .	Principal Id, Null
peristiwa:eventBusInvocation	`"events:eventBusInvocation":" boolean"` Untuk *boolean*, gunakan BETUL ketika aturan mengirimkan peristiwa ke target yang merupakan bus peristiwa di akun lain. Gunakan SALAH ketika ketika `PutEvents` panggilan API digunakan.	eventBusInvocation, Null
peristiwa:ManagedBy	Digunakan secara internal oleh AWS layanan. Untuk aturan yang dibuat oleh AWS layanan atas nama Anda, nilainya adalah nama utama layanan yang membuat aturan.	Tidak dimaksudkan untuk digunakan dalam kebijakan pelanggan.

Kunci syarat	Pasangan nilai kunci	Jenis evaluasi
events:source	`"events:source":" source "` Gunakan *`sumber`* untuk string literal untuk bidang sumber peristiwa seperti`"aws.ec2"` atau `"aws.s3"`. Untuk nilai yang lebih mungkin untuk *`sumber`*, lihat contoh peristiwa di Acara dari AWS layanan.	Source, Null
peristiwa:TargetArn	`"events:TargetArn" :" target-arn "` Untuk *`target-arn`* , gunakan ARN target untuk aturan, misalnya`"arn:aws:lambda:*:*:function:*"` .	ArrayOfARN, Null

Misalnya pernyataan kebijakan untuk EventBridge, lihatMengelola izin akses keEventBridge sumber daya Amazon Anda.

Topik

- EventBridge Pipa spesifik
- Contoh: Menggunakan creatorAccount syarat
- Contoh: Menggunakan eventBusInvocation syarat
- Contoh: Membatasi akses ke sumber tertentu
- Contoh: Mendefinisikan beberapa sumber yang dapat digunakan dalam pola peristiwa individual
- Contoh: Mendefinisikan sumber dan DetailType yang dapat digunakan dalam pola peristiwa
- Contoh: Memastikan bahwa sumber didefinisikan dalam pola peristiwa
- Contoh: Mendefinisikan daftar sumber yang diperbolehkan dalam pola peristiwa dengan beberapa sumber

- Contoh: Membatasi PutRule akses oleh detail.service
- Contoh: Membatasi PutRule akses oleh detail.eventTypeCode
- Contoh: Memastikan bahwa hanya AWS CloudTrail peristiwa untuk panggilan API dari PrincipalId yang diizinkan
- Contoh: Membatasi akses ke target

EventBridge Pipa spesifik

EventBridge Pipa tidak mendukung kunci kondisi kebijakan IAM tambahan.

Contoh: Menggunakan **`creatorAccount`** syarat

Contoh pernyataan kebijakan berikut menunjukkan cara menggunakan `creatorAccount` syarat dalam kebijakan untuk hanya mengizinkan aturan yang akan dibuat jika akun ditetapkan sebagai`creatorAccount` adalah akun yang membuat aturan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleForOwnedRules",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "events:creatorAccount": "${aws:PrincipalAccount}"
                }
            }
        }
    ]
}
```

Contoh: Menggunakan **`eventBusInvocation`** syarat

`eventBusInvocation` menunjukkan apakah permintaan berasal dari target lintas-akun atau `PutEvents` permintaan API. Nilai adalah BETUL ketika hasil permintaan dari aturan yang mencakup target lintas-akun, seperti ketika target adalah bus peristiwa di akun lain. Nilai adalah SALAH ketika hasil permintaan dari `PutEvents` permintaan API. Contoh berikut menunjukkan permintaan dari target lintas-akun.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCrossAccountInvocationEventsOnly",
      "Effect": "Allow",
      "Action": "events:PutEvents",
      "Resource": "*",
      "Condition": {
        "BoolIfExists": {
          "events:eventBusInvocation": "true"
        }
      }
    }
  ]
}
```

Contoh: Membatasi akses ke sumber tertentu

Contoh kebijakan berikut dapat disematkan ke pengguna IAM. Kebijakan A memungkinkan `PutRule` tindakan API untuk semua peristiwa, sementara Kebijakan B memungkinkan `PutRule` hanya jika pola peristiwa aturan yang dibuat cocok dengan peristiwa Amazon EC2.

Kebijakan A: memungkinkan semua peristiwa

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleForAllEvents",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*"
        }
    ]
    }
```

Kebijakan B: —memungkinkan peristiwa hanya dari Amazon EC2

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "AllowPutRuleForAllEC2Events",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:source": "aws.ec2"
                }
            }
        }
    ]
}
```

`EventPattern` adalah argumen wajib untuk `PutRule`. Oleh karena itu, jika pengguna dengan Kebijakan B memanggil `PutRule`dengan pola peristiwa seperti berikut ini.

```
{
    "source": [ "aws.ec2" ]
}
```

Aturan akan dibuat karena kebijakan memungkinkan untuk sumber khusus ini: yaitu, `"aws.ec2"`. Namun, jika pengguna dengan Kebijakan B memanggil `PutRule` dengan pola peristiwa seperti berikut ini, pembuatan aturan akan ditolak karena kebijakan tidak mengizinkan sumber spesifik ini: yaitu, `"aws.s3"`.

```
{
    "source": [ "aws.s3" ]
}
```

Pada dasarnya, pengguna dengan Kebijakan B hanya diperbolehkan untuk membuat aturan yang akan cocok dengan peristiwa yang berasal dari Amazon EC2; maka, mereka hanya diperbolehkan mengakses ke peristiwa dari Amazon EC2.

Lihat tabel berikut untuk perbandingan Kebijakan A dan Kebijakan B.

Pola Peristiwa	Diperbolehkan oleh Kebijakan A	Diperbolehkan oleh Kebijakan B
`{`	Ya	Ya

Pola Peristiwa	Diperbolehkan oleh Kebijakan A	Diperbolehkan oleh Kebijakan B
`{ "source": [ "aws.ec2" ] }`		
`{ "source": [ "aws.ec2", "aws.s3" ] }`	Ya	Tidak (Sumber aws.s3 tidak diizinkan)
`{ "source": [ "aws.ec2" ], "detail-type": [ "EC2 Instance State-change Notification" ] }`	Ya	Ya
`{ "detail-type": [ "EC2 Instance State-change Notification" ] }`	Ya	Tidak (Sumber harus ditentukan)

Contoh: Mendefinisikan beberapa sumber yang dapat digunakan dalam pola peristiwa individual

Kebijakan berikut memungkinkan pengguna atau peran IAM untuk membuat aturan di mana sumber dalam `EventPattern` adalah Amazon EC2 atau Amazon ECS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleIfSourceIsEC2OrECS",
```

```
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:source": [ "aws.ec2", "aws.ecs" ]
                }
            }
        }
    ]
}
```

Tabel berikut menunjukkan beberapa contoh pola peristiwa yang diizinkan atau ditolak oleh kebijakan ini.

Pola peristiwa	Diizinkan oleh kebijakan
`{ "source": [ "aws.ec2" ] }`	Ya
`{ "source": [ "aws.ecs" ] }`	Ya
`{ "source": [ "aws.s3" ] }`	Tidak
`{ "source": [ "aws.ec2", "aws.ecs" ] }`	Tidak
`{ "detail-type": [ "AWS API Call via CloudTrail" ] }`	Tidak

Contoh: Mendefinisikan sumber dan **`DetailType`** yang dapat digunakan dalam pola peristiwa

Kebijakan berikut memungkinkan peristiwa hanya dari `aws.ec2` sumber dengan `DetailType` sama dengan `EC2 instance state change notification`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid":
 "AllowPutRuleIfSourceIsEC2AndDetailTypeIsInstanceStateChangeNotification",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:source": "aws.ec2",
                    "events:detail-type": "EC2 Instance State-change Notification"
                }
            }
        }
    ]
}
```

Tabel berikut menunjukkan beberapa contoh pola peristiwa yang diizinkan atau ditolak oleh kebijakan ini.

Pola peristiwa	Diizinkan oleh kebijakan
`{ "source": [ "aws.ec2" ] }`	Tidak
`{ "source": [ "aws.ecs" ] }`	Tidak
`{`	Ya

Pola peristiwa	Diizinkan oleh kebijakan

<pre>"source": ["aws.ec2"],
"detail-type": ["EC2
Instance State-change Notificat
ion"]
}</pre>	

<pre>{
"source": ["aws.ec2"],
"detail-type": ["EC2
Instance Health Failed"]
}</pre>
	Tidak

<pre>{
"detail-type": ["EC2
Instance State-change Notificat
ion"]
}</pre>
	Tidak

Contoh: Memastikan bahwa sumber didefinisikan dalam pola peristiwa

Kebijakan berikut memungkinkan pengguna untuk hanya membuat aturan dengan `EventPatterns` yang memiliki bidang sumber. Dengan kebijakan ini, pengguna atau peran IAM tidak dapat membuat aturan dengan `EventPattern` yang tidak menyediakan sumber tertentu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleIfSourceIsSpecified",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "Null": {
                    "events:source": "false"
                }
            }
        }
```

```
    ]
}
```

Tabel berikut menunjukkan beberapa contoh pola peristiwa yang diizinkan atau ditolak oleh kebijakan ini.

Pola Peristiwa	Diperbolehkan oleh Kebijakan
`{ "source": [ "aws.ec2" ], "detail-type": [ "EC2 Instance State-change Notification" ] }`	Ya
`{ "source": [ "aws.ecs", "aws.ec2" ] }`	Ya
`{ "detail-type": [ "EC2 Instance State-change Notification" ] }`	Tidak

Contoh: Mendefinisikan daftar sumber yang diperbolehkan dalam pola peristiwa dengan beberapa sumber

Kebijakan berikut memungkinkan pengguna untuk membuat aturan dengan `EventPatterns` yang memiliki banyak sumber di dalamnya. Setiap sumber dalam pola peristiwa harus menjadi anggota dari daftar yang disediakan dalam syarat tersebut. Ketika Anda menggunakan`ForAllValues` syarat, pastikan bahwa setidaknya salah satu item dalam daftar syarat didefinisikan.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "AllowPutRuleIfSourceIsSpecifiedAndIsEitherS3OrEC2OrBoth",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "events:source": [ "aws.ec2", "aws.s3" ]
                },
                "Null": {
                    "events:source": "false"
                }
            }
        }
    ]
}
```

Tabel berikut menunjukkan beberapa contoh pola peristiwa yang diizinkan atau ditolak oleh kebijakan ini.

Pola Peristiwa	Diperbolehkan oleh Kebijakan
`{ "source": [ "aws.ec2" ] }`	Ya
`{ "source": [ "aws.ec2", "aws.s3" ] }`	Ya
`{ "source": [ "aws.ec2", "aws.autoscaling" ] }`	Tidak
`{ "detail-type": [ "EC2 Instance State-change Notification" ]`	Tidak

Pola Peristiwa	Diperbolehkan oleh Kebijakan
}	

Contoh: Membatasi **`PutRule`** akses oleh **`detail.service`**

Anda dapat membatasi pengguna atau peran IAM untuk membuat aturan hanya untuk peristiwa yang memiliki nilai tertentu dalam `events:details.service`bidang. Nilai dari `events:details.service` belum tentu nama AWS layanan.

Syarat kebijakan ini sangat membantu ketika Anda bekerja dengan peristiwa dari AWS Health yang berhubungan dengan keamanan atau penyalahgunaan. Dengan menggunakan syarat kebijakan ini, Anda dapat membatasi akses ke peringatan sensitif ini hanya untuk pengguna yang perlu melihatnya.

Sebagai contoh, kebijakan berikut memungkinkan pembuatan aturan hanya untuk peristiwa di mana nilai `events:details.service` adalah `ABUSE`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleEventsWithDetailServiceEC2",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:detail.service": "ABUSE"
                }
            }
        }
    ]
}
```

Contoh: Membatasi **`PutRule`** akses oleh **`detail.eventTypeCode`**

Anda dapat membatasi pengguna atau peran IAM untuk membuat aturan hanya untuk peristiwa yang memiliki nilai tertentu dalam `events:details.eventTypeCode`bidang. Syarat kebijakan ini sangat membantu ketika Anda bekerja dengan peristiwa dari AWS Health yang berhubungan dengan

keamanan atau penyalahgunaan. Dengan menggunakan syarat kebijakan ini, Anda dapat membatasi akses ke peringatan sensitif ini hanya untuk pengguna yang perlu melihatnya.

Sebagai contoh, kebijakan berikut memungkinkan pembuatan aturan hanya untuk peristiwa di mana nilai `events:details.eventTypeCode` adalah `AWS_ABUSE_DOS_REPORT`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleEventsWithDetailServiceEC2",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:detail.eventTypeCode": "AWS_ABUSE_DOS_REPORT"
                }
            }
        }
    ]
}
```

Contoh: Memastikan bahwa hanya AWS CloudTrail peristiwa untuk panggilan API dari **`PrincipalId`** yang diizinkan

SemuaAWS CloudTrail peristiwa memiliki pengguna yang membuat panggilan API di`detail.userIdentity.principalId` jalur peristiwa. PrincipalId Menggunakan kunci`events:detail.userIdentity.principalId` syarat, Anda dapat membatasi akses pengguna atau peran IAM ke CloudTrail peristiwa hanya untuk pengguna yang berasal dari akun tertentu.

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleOnlyForCloudTrailEventsWhereUserIsASpecificIAMUser",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
```

```
                "StringEquals": {
                    "events:detail-type": [ "AWS API Call via CloudTrail" ],
                    "events:detail.userIdentity.principalId":
 [ "AIDAJ45Q7YFFAREXAMPLE" ]
                }
            }
        }
    ]
}
```

Tabel berikut menunjukkan beberapa contoh pola peristiwa yang diizinkan atau ditolak oleh kebijakan ini.

Pola peristiwa	Diizinkan oleh kebijakan
`{ "detail-type": [ "AWS API Call via CloudTrail" ] }`	Tidak
`{ "detail-type": [ "AWS API Call via CloudTrail" ], "detail.userIdentity.principalId": [ "AIDAJ45Q7YFFAREXAMPLE" ] }`	Ya
`{ "detail-type": [ "AWS API Call via CloudTrail" ], "detail.userIdentity.principalId": [ "AROAIDPPEZS35WEXAMPLE:AssumedRoleSessionName" ] }`	Tidak

Contoh: Membatasi akses ke target

Jika pengguna atau peran IAM memiliki `events:PutTargets` izin, mereka dapat menambahkan target apa pun dalam akun yang sama dengan aturan yang diizinkan untuk diakses. Kebijakan berikut membatasi pengguna untuk menambahkan target hanya untuk aturan tertentu: `MyRule` dalam akun `123456789012`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutTargetsOnASpecificRule",
            "Effect": "Allow",
            "Action": "events:PutTargets",
            "Resource": "arn:aws:events:us-east-1:123456789012:rule/MyRule"
        }
    ]
}
```

Untuk membatasi target apa yang dapat ditambahkan ke aturan, gunakan `events:TargetArn` kunci syarat. Anda dapat membatasi target hanya untuk fungsi Lambda, seperti dalam contoh berikut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutTargetsOnASpecificRuleAndOnlyLambdaFunctions",
            "Effect": "Allow",
            "Action": "events:PutTargets",
            "Resource": "arn:aws:events:us-east-1:123456789012:rule/MyRule",
            "Condition": {
                "ArnLike": {
                    "events:TargetArn": "arn:aws:lambda:*:*:function:*"
                }
            }
        }
    ]
}
```

Menggunakan peran terkait layanan untuk EventBridge

Amazon EventBridge menggunakan AWS Identity and Access Management peran yang terkait dengan layanan (IAM) . Peran yang terkait dengan layanan adalah tipe IAM role unik yang terkait langsung ke layanan. Peran yang terkait dengan layanan ditentukan sebelumnya oleh EventBridge dan meliputi semua izin yang diperlukan layanan untuk menghubungi layanan AWS lainnya atas nama Anda.

Topik

- Menggunakan peran untuk membuat rahasia untuk tujuan API
- Menggunakan peran untuk penemuan skema

Menggunakan peran untuk membuat rahasia untuk tujuan API

Amazon EventBridge menggunakan AWS Identity and Access Management peran yang terkait dengan layanan (IAM) . Peran yang terkait dengan layanan adalah tipe IAM role unik yang terkait langsung ke layanan. Peran yang terhubung dengan layanan ditentukan sebelumnya oleh EventBridge dan mencakup semua izin yang diperlukan layanan untuk menghubungi layanan AWS lainnya atas nama Anda.

Peran yang terhubung dengan layanan memudahkan pengaturan EventBridge karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. EventBridge menentukan izin peran yang terhubung dengan layanan, dan kecuali ditentukan sebaliknya, hanya EventBridge yang dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran yang terhubung dengan layanan hanya setelah menghapus sumber daya terkait terlebih dahulu. Ini melindungi sumber daya EventBridge karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, silakan lihat layanan AWS yang bisa digunakan dengan IAM dan carilah layanan yang memiliki opsi Ya di kolom Peran terkait layanan. Pilih Ya dengan sebuah tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Izin peran terkait layanan untuk EventBridge

EventBridgemenggunakan peran terkait layanan bernama AWSServiceRoleForAmazonEventBridgeApiDestinations— Memungkinkan akses ke Rahasia Secrets Manager yang dibuat oleh. EventBridge

Peran tertaut layanan AWSServiceRoleForAmazonEventBridgeApiDestinations memercayai layanan berikut untuk mengambil peran tersebut:

- `apidestinations.events.amazonaws.com`

Kebijakan izin peran bernama AmazonEventBridgeApiDestinationsServiceRolePolicy memungkinkan EventBridge untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- Tindakan: `create, describe, update and delete secrets; get and put secret values` pada `secrets created for all connections by EventBridge`

Anda harus mengonfigurasi izin agar pengguna, grup, atau peran Anda membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, lihat Izin peran yang terkait dengan layanan dalam Panduan Pengguna IAM.

Membuat peran yang terkait dengan layanan untuk EventBridge

Anda tidak perlu membuat peran tertaut layanan secara manual. Saat Anda membuat koneksi diAWS Management Console, APIAWS CLI, atau AWS API, EventBridge buat peran terkait layanan untuk Anda.

> **Important**
>
> Peran tertaut layanan ini dapat muncul di akun Anda jika Anda menyelesaikan tindakan di layanan lain yang menggunakan fitur yang disupport oleh peran ini. Jika Anda menggunakan EventBridge layanan sebelum 11 Februari 2021, ketika mulai mendukung peran terkait layanan, maka EventBridge buat AWSServiceRoleForAmazonEventBridgeApiDestinationsperan tersebut di akun Anda. Untuk mempelajari lebih lanjut, lihat Peran baru muncul di saya Akun AWS.

Jika Anda menghapus peran terkait layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Saat Anda membuat koneksi, EventBridge buat peran terkait layanan untuk Anda lagi.

Mengedit peran terkait layanan untuk EventBridge

EventBridge tidak mengizinkan Anda untuk mengedit peran tertaut layanan AWSServiceRoleForAmazonEventBridgeApiDestinations. Setelah membuat peran tertaut layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengedit deskripsi peran menggunakan IAM. Untuk informasi lebih lanjut, lihat Mengedit peran terkait layanan dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk EventBridge

Jika Anda tidak lagi perlu menggunakan fitur atau layanan yang memerlukan peran tertaut layanan, sebaiknya hapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas yang tidak digunakan dan tidak dipantau atau dipelihara secara aktif. Namun, Anda harus membersihkan peran tertaut layanan terlebih dahulu sebelum dapat menghapusnya secara manual.

Membersihkan peran tertaut-layanan

Sebelum dapat menggunakan IAM untuk menghapus peran tertaut-layanan, Anda harus terlebih dahulu menghapus semua sumber daya yang digunakan oleh peran tersebut.

> **Note**
>
> Jika layanan EventBridge menggunakan peran tersebut ketika Anda mencoba menghapus sumber daya, penghapusan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk menghapus EventBridge sumber daya yang digunakan oleh AWSServiceRoleForAmazonEventBridgeApiDestinations(konsol)

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di bawah Integrasi pilih tujuan API, lalu pilih tab Koneksi.
3. Pilih koneksi, lalu pilih Hapus.

Untuk menghapus EventBridge sumber daya yang digunakan oleh AWSServiceRoleForAmazonEventBridgeApiDestinations(AWSCLI)

- Gunakan perintah berikut:`delete-connection`.

Untuk menghapus EventBridge sumber daya yang digunakan oleh AWSServiceRoleForAmazonEventBridgeApiDestinations(API)

- Gunakan perintah berikut:`DeleteConnection`.

Menghapus peran tertaut layanan secara manual

Gunakan konsol IAM, AWS CLI, atau AWS API untuk menghapus peran terkait layanan AWSServiceRoleForAmazonEventBridgeApiDestinations Untuk informasi lebih lanjut, lihat Menghapus peran terkait layanan dalam Panduan Pengguna IAM.

Wilayah yang didukung untuk peran yang terhubung dengan layanan EventBridge

EventBridge memberikan dukungan dengan peran yang terhubung dengan layanan di semua Wilayah tempat layanan tersedia. Untuk informasi selengkapnya, lihat AWSWilayah dan titik akhir.

Menggunakan peran untuk penemuan skema

Amazon EventBridge menggunakan AWS Identity and Access Management peran yang terkait dengan layanan (IAM) . Peran yang terkait dengan layanan adalah tipe IAM role unik yang terkait langsung ke layanan. Peran yang terhubung dengan layanan ditentukan sebelumnya oleh EventBridge dan mencakup semua izin yang diperlukan layanan untuk menghubungi layanan AWS lainnya atas nama Anda.

Peran yang terhubung dengan layanan memudahkan pengaturan EventBridge karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. EventBridge menentukan izin peran yang terhubung dengan layanan, dan kecuali ditentukan sebaliknya, hanya EventBridge yang dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran yang terhubung dengan layanan hanya setelah menghapus sumber daya terkait terlebih dahulu. Ini melindungi sumber daya EventBridge karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, silakan lihat layanan AWS yang bisa digunakan dengan IAM dan carilah layanan yang memiliki opsi Ya di kolom Peran terkait layanan. Pilih Ya dengan sebuah tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Izin peran terkait layanan untuk EventBridge

EventBridgemenggunakan peran terkait layanan bernama AWSServiceRoleForSchemas—Memberikan izin untuk Aturan Terkelola yang dibuat oleh skema.. Amazon EventBridge

Peran tertaut layanan AWSServiceRoleForSchemas memercayai layanan berikut untuk mengambil peran tersebut:

- `schemas.amazonaws.com`

Kebijakan izin peran bernama AmazonEventBridgeSchemasServiceRolePolicymemungkinkan EventBridge untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- Tindakan: `put, enable, disable, and delete rules; put and remove targets; list targets per rule` pada `all managed rules created by EventBridge`

Anda harus mengonfigurasi izin agar pengguna, grup, atau peran Anda membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, lihat Izin peran yang terkait dengan layanan dalam Panduan Pengguna IAM.

Membuat peran yang terkait dengan layanan untuk EventBridge

Anda tidak perlu membuat peran tertaut layanan secara manual. Saat Anda menjalankan Schema Discovery diAWS Management Console, theAWS CLI, atau AWS API, EventBridge buat peran terkait layanan untuk Anda.

> **Important**
>
> Peran tertaut layanan ini dapat muncul di akun Anda jika Anda menyelesaikan tindakan di layanan lain yang menggunakan fitur yang disupport oleh peran ini. Jika Anda menggunakan EventBridge layanan sebelum 27 November 2019, ketika mulai mendukung peran terkait layanan, maka EventBridge buat AWSServiceRoleForSchemasperan di akun Anda. Untuk mempelajari lebih lanjut, lihat Peran baru muncul di saya Akun AWS.

Jika Anda menghapus peran terkait layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Saat Anda melakukan Penemuan Skema, EventBridge buat peran terkait layanan untuk Anda lagi.

Mengedit peran terkait layanan untuk EventBridge

EventBridge tidak mengizinkan Anda untuk mengedit peran tertaut layanan AWSServiceRoleForSchemas. Setelah membuat peran tertaut layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengedit deskripsi peran menggunakan IAM. Untuk informasi lebih lanjut, lihat Mengedit peran terkait layanan dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk EventBridge

Jika Anda tidak lagi perlu menggunakan fitur atau layanan yang memerlukan peran tertaut layanan, sebaiknya hapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas yang tidak digunakan dan tidak dipantau atau dipelihara secara aktif. Namun, Anda harus membersihkan peran tertaut layanan terlebih dahulu sebelum dapat menghapusnya secara manual.

Membersihkan peran tertaut-layanan

Sebelum dapat menggunakan IAM untuk menghapus peran tertaut-layanan, Anda harus terlebih dahulu menghapus semua sumber daya yang digunakan oleh peran tersebut.

> **Note**
>
> Jika layanan EventBridge menggunakan peran tersebut ketika Anda mencoba menghapus sumber daya, penghapusan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk menghapus EventBridge sumber daya yang digunakan oleh AWSServiceRoleForSchemas(konsol)

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Di bawah Bus pilih bus Event, lalu pilih bus acara.
3. Pilih Hentikan penemuan.

Untuk menghapus EventBridge sumber daya yang digunakan oleh AWSServiceRoleForSchemas(AWSCLI)

- Gunakan perintah berikut:`delete-discoverer`.

Untuk menghapus EventBridge sumber daya yang digunakan oleh AWSServiceRoleForSchemas(API)

- Gunakan perintah berikut:`DeleteDiscoverer`.

Menghapus peran tertaut layanan secara manual

Gunakan konsol IAM, AWS CLI, atau AWS API untuk menghapus peran terkait layanan AWSServiceRoleForSchemas Untuk informasi lebih lanjut, lihat Menghapus peran terkait layanan dalam Panduan Pengguna IAM.

Wilayah yang didukung untuk peran yang terhubung dengan layanan EventBridge

EventBridge memberikan dukungan dengan peran yang terhubung dengan layanan di semua Wilayah tempat layanan tersedia. Untuk informasi selengkapnya, lihat AWSWilayah dan titik akhir.

Logging panggilan Amazon EventBridge API menggunakan AWS CloudTrail

Amazon EventBridge terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau Layanan AWS. CloudTrail menangkap semua panggilan API untuk EventBridge sebagai peristiwa. Panggilan yang diambil termasuk panggilan dari EventBridge konsol dan panggilan kode ke operasi EventBridge API. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat EventBridge, alamat IP dari mana permintaan dibuat, kapan dibuat, dan detail tambahan.

Setiap peristiwa atau entri log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan berikut hal ini:

- Baik permintaan tersebut dibuat dengan kredensial pengguna root atau pengguna.
- Apakah permintaan dibuat atas nama pengguna IAM Identity Center.
- Apakah permintaan dibuat dengan kredensial keamanan sementara untuk suatu peran atau pengguna gabungan.
- Apakah permintaan tersebut dibuat oleh Layanan AWS lain.

CloudTrail aktif di Anda Akun AWS ketika Anda membuat akun dan Anda secara otomatis memiliki akses ke riwayat CloudTrail Acara. Riwayat CloudTrail Acara menyediakan catatan yang dapat dilihat, dapat dicari, dapat diunduh, dan tidak dapat diubah dari 90 hari terakhir dari peristiwa manajemen yang direkam dalam file. Wilayah AWS Untuk informasi selengkapnya, lihat Bekerja dengan riwayat CloudTrail Acara di Panduan AWS CloudTrail Pengguna. Tidak ada CloudTrail biaya untuk melihat riwayat Acara.

Untuk catatan acara yang sedang berlangsung dalam 90 hari Akun AWS terakhir Anda, buat jejak atau penyimpanan data acara CloudTrailDanau.

CloudTrail jalan setapak

> Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Semua jalur yang dibuat menggunakan AWS Management Console Multi-region. Anda dapat membuat jalur Single-region atau Multi-region dengan menggunakan. AWS CLI Membuat jejak Multi-wilayah disarankan karena Anda menangkap aktivitas Wilayah AWS di semua akun Anda. Jika Anda membuat jejak wilayah Tunggal, Anda hanya dapat melihat peristiwa yang dicatat di jejak.

Wilayah AWS Untuk informasi selengkapnya tentang jejak, lihat Membuat jejak untuk Anda Akun AWS dan Membuat jejak untuk organisasi di Panduan AWS CloudTrail Pengguna.

Anda dapat mengirimkan satu salinan acara manajemen yang sedang berlangsung ke bucket Amazon S3 Anda tanpa biaya CloudTrail dengan membuat jejak, namun, ada biaya penyimpanan Amazon S3. Untuk informasi selengkapnya tentang CloudTrail harga, lihat AWS CloudTrail Harga. Untuk informasi tentang harga Amazon S3, lihat Harga Amazon S3.

CloudTrail Menyimpan data acara danau

CloudTrail Lake memungkinkan Anda menjalankan kueri berbasis SQL pada acara Anda. CloudTrail Lake mengonversi peristiwa yang ada dalam format JSON berbasis baris ke format Apache ORC. ORC adalah format penyimpanan kolumnar yang dioptimalkan untuk pengambilan data dengan cepat. Peristiwa digabungkan ke dalam penyimpanan data peristiwa, yang merupakan kumpulan peristiwa yang tidak dapat diubah berdasarkan kriteria yang Anda pilih dengan menerapkan pemilih acara tingkat lanjut. Penyeleksi yang Anda terapkan ke penyimpanan data acara mengontrol peristiwa mana yang bertahan dan tersedia untuk Anda kueri. Untuk informasi lebih lanjut tentang CloudTrail Danau, lihat Bekerja dengan AWS CloudTrail Danau di Panduan AWS CloudTrail Pengguna.

CloudTrail Penyimpanan data acara danau dan kueri menimbulkan biaya. Saat Anda membuat penyimpanan data acara, Anda memilih opsi harga yang ingin Anda gunakan untuk penyimpanan data acara. Opsi penetapan harga menentukan biaya untuk menelan dan menyimpan peristiwa, dan periode retensi default dan maksimum untuk penyimpanan data acara. Untuk informasi selengkapnya tentang CloudTrail harga, lihat AWS CloudTrail Harga.

EventBridge peristiwa data di CloudTrail

Peristiwa data memberikan informasi tentang operasi sumber daya yang dilakukan pada atau di sumber daya (misalnya, membaca atau menulis ke objek Amazon S3). Ini juga dikenal sebagai operasi bidang data. Peristiwa data seringkali merupakan aktivitas volume tinggi. Secara default, CloudTrail tidak mencatat peristiwa data. Riwayat CloudTrail peristiwa tidak merekam peristiwa data.

Biaya tambahan berlaku untuk peristiwa data. Untuk informasi selengkapnya tentang CloudTrail harga, lihat AWS CloudTrail Harga.

Anda dapat mencatat peristiwa data untuk jenis EventBridge sumber daya menggunakan CloudTrail konsol AWS CLI, atau operasi CloudTrail API. Untuk informasi selengkapnya tentang cara mencatat

peristiwa data, lihat Mencatat peristiwa data dengan AWS Management Console dan Logging peristiwa data dengan AWS Command Line Interface di Panduan AWS CloudTrail Pengguna.

Tabel berikut mencantumkan jenis EventBridge sumber daya yang dapat Anda log peristiwa data. Kolom tipe peristiwa data (konsol) menunjukkan nilai yang akan dipilih dari daftar tipe peristiwa Data di CloudTrail konsol. Kolom nilai resources.type menunjukkan **`resources.type`** nilai, yang akan Anda tentukan saat mengonfigurasi penyeleksi acara lanjutan menggunakan API atau. AWS CLI CloudTrail CloudTrailKolom API Data yang dicatat ke menampilkan panggilan API yang dicatat CloudTrail untuk jenis sumber daya.

Jenis peristiwa data (konsol)	nilai resources.type	API data masuk CloudTrail
Bus acara	`AWS::Events::Event Bus`	• DescribeEventBus
Aturan bus acara	`AWS::Events::Rule`	• DeleteRule • DescribeRule • DisableRule • EnableRule • ListRuleNamesByTarget • ListRules • ListTargetsByRule • PutRule • PutTargets • RemoveTargets • TestEventPattern
Pipa	`AWS::Pipes::Pipe`	• CreatePipe • DeletePipe • DescribePipe • ListPipes • StartPipe • StopPipe • UpdatePipe

Anda dapat mengonfigurasi pemilih acara lanjutan untuk memfilter pada `eventName`,`readOnly`, dan `resources.ARN` bidang untuk mencatat hanya peristiwa yang penting bagi Anda. Untuk informasi selengkapnya tentang bidang ini, lihat AdvancedFieldSelectordi Referensi AWS CloudTrail API.

EventBridge acara manajemen di CloudTrail

Acara manajemen memberikan informasi tentang operasi manajemen yang dilakukan pada sumber daya di Anda Akun AWS. Ini juga dikenal sebagai operasi pesawat kontrol. Secara default, CloudTrail mencatat peristiwa manajemen.

Amazon EventBridge mencatat semua operasi pesawat EventBridge kontrol sebagai peristiwa manajemen. Untuk daftar operasi bidang Amazon EventBridge kontrol yang EventBridge masuk ke log CloudTrail, lihat Referensi Amazon EventBridge API.

EventBridge contoh acara

Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang operasi API yang diminta, tanggal dan waktu operasi, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, sehingga peristiwa tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan CloudTrail peristiwa yang menunjukkan `PutRule` operasi.

```
{
  "eventVersion":"1.03",
  "userIdentity":{
    "type":"Root",
    "principalId":"123456789012",
    "arn":"arn:aws:iam::123456789012:root",
    "accountId":"123456789012",
    "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
    "sessionContext":{
      "attributes":{
      "mfaAuthenticated":"false",
      "creationDate":"2015-11-17T23:56:15Z"
      }
    }
  },
  "eventTime":"2015-11-18T00:11:28Z",
  "eventSource":"events.amazonaws.com",
  "eventName":"PutRule",
```

```
  "awsRegion":"us-east-1",
  "sourceIPAddress":"AWS Internal",
  "userAgent":"AWS CloudWatch Console",
  "requestParameters":{
    "description":"",
    "name":"cttest2",
    "state":"ENABLED",
    "eventPattern":"{\"source\":[\"aws.ec2\"],\"detail-type\":[\"EC2 Instance State-
change Notification\"]}",
    "scheduleExpression":""
  },
  "responseElements":{
    "ruleArn":"arn:aws:events:us-east-1:123456789012:rule/cttest2"
  },
  "requestID":"e9caf887-8d88-11e5-a331-3332aa445952",
  "eventID":"49d14f36-6450-44a5-a501-b0fdcdfaeb98",
  "eventType":"AwsApiCall",
  "apiVersion":"2015-10-07",
  "recipientAccountId":"123456789012"
}
```

Untuk informasi tentang konten CloudTrail rekaman, lihat konten CloudTrail rekaman di Panduan AWS CloudTrail Pengguna.

CloudTrail entri log untuk tindakan yang diambil oleh Pipes EventBridge

EventBridge Pipes mengasumsikan peran IAM yang disediakan saat membaca peristiwa dari sumber, memohon pengayaan, atau memanggil target. Untuk CloudTrail entri yang terkait dengan tindakan yang dilakukan di akun Anda pada semua pengayaan, target, dan sumber Amazon SQS, Kinesis, dan DynamoDB, bidang dan akan disertakan. `sourceIPAddress` `invokedBy` `pipes.amazonaws.com`

Contoh entri CloudTrail log untuk semua sumber pengayaan, target, dan Amazon SQS, Kinesis, dan DynamoDB

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "...",
    "arn": "arn:aws:sts::111222333444:assumed-role/...",
    "accountId": "111222333444",
```

```
    "accessKeyId": "...",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "...",
        "arn": "...",
        "accountId": "111222333444",
        "userName": "userName"
      },
      "webIdFederationData": {},
      "attributes": {
        "creationDate": "2022-09-22T21:41:15Z",
        "mfaAuthenticated": "false"
      }
    },
    "invokedBy": "pipes.amazonaws.com"
  },
  "eventTime": ",,,",
  "eventName": "...",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "pipes.amazonaws.com",
  "userAgent": "pipes.amazonaws.com",
  "requestParameters": {
    ...
  },
  "responseElements": null,
  "requestID": "...",
  "eventID": "...",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "...",
  "eventCategory": "Management"
}
```

Untuk semua sumber lain, `sourceIPAddress` bidang entri CloudTrail log akan memiliki alamat IP dinamis dan tidak boleh diandalkan untuk integrasi atau kategorisasi peristiwa apa pun. Selain itu, entri ini tidak akan memiliki `invokedBy` bidang.

Contoh entri CloudTrail log untuk semua sumber lain

```
{
  "eventVersion": "1.08",
```

```
  "userIdentity": {
    ...
  },
  "eventTime": ",,,",
  "eventName": "...",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "Python-httplib2/0.8 (gzip)",
}
```

Validasi kepatuhan di Amazon EventBridge

Auditor pihak ke tiga seperti SOC, PCI, FedRAMP, dan HIPAA menilai keamanan dan kepatuhan layanan AWS sebagai bagian dari beberapa program kepatuhan AWS.

Untuk daftar layanan AWS dalam cakupan program kepatuhan tertentu, lihat AWS Layanan dalam Cakupan menurut Program Kepatuhan. Untuk informasi umum, lihat Program Kepatuhan AWS.

Anda bisa mengunduh laporan audit pihak ke tiga menggunakan AWS Artifact. Untuk informasi lebih lanjut, lihat Mengunduh Laporan di AWS Artifact.

Tanggung jawab kepatuhan Anda saat menggunakan EventBridge ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, serta undang-undang dan peraturan yang berlaku. AWSmenyediakan sumber daya berikut untuk membantu kepatuhan:

- Panduan Mulai Cepat Keamanan dan Kepatuhan – Pertimbangan arsitektur dan langkah-langkah untuk menerapkan lingkungan dasar yang berfokus pada keamanan dan kepatuhan pada AWS.
- perancangan laporan resmi keamanan dan kepatuhan HIPAA – Bagaimana perusahaan dapat menggunakan AWS untuk membuat aplikasi yang sesuai dengan HIPAA.
- AWS Sumber kepatuhan – Koleksi buku kerja dan panduan.
- Mengevaluasi Sumber Daya dengan Aturan di AWS ConfigPanduan Developer – Informasi tentang cara AWS Config menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- AWS Security Hub – Pandangan komprehensif tentang status keamanan Anda di dalam AWS yang membantu Anda memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik.

Ketahanan Amazon EventBridge

AWSInfrastruktur global dibangun di sekitar AWS Wilayah dan Availability Zones AWS Wilayah menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan jaringan berlatensi rendah, throughput yang tinggi, dan sangat redundan. Dengan Availability Zone, Anda dapat mendesain dan mengoperasikan aplikasi dan basis data yang secara otomatis mengalami kegagalan 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 Infrastruktur Global AWS.

Keamanan infrastruktur di AmazonEventBridge

Sebagai layanan terkelola, AmazonEventBridgedilindungi olehAWSkeamanan jaringan global. Untuk informasi tentangAWSlayanan keamanan dan bagaimanaAWSmelindungi infrastruktur, lihatAWSKeamanan Cloud. Untuk mendesainAWSlingkungan menggunakan praktik terbaik untuk keamanan infrastruktur, lihatPerlindungan InfrastrukturdiPilar KeamananAWSKerangka Kerja yang Diarsiteksikan dengan Baik.

Anda menggunakan panggilan API AWS yang dipublikasikan untuk mengakses EventBridge melalui jaringan. Klien harus mendukung hal berikut:

- Transport Layer Security (TLS). Kami membutuhkan TLS 1.2 dan merekomendasikan TLS 1.3.
- Suite cipher dengan kerahasiaan maju sempurna (PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan sistem yang lebih baru mendukung mode ini.

Selain itu, permintaan harus ditandatangani menggunakan access key ID dan secret access key yang terkait dengan principal IAM. Atau Anda bisa menggunakan AWS Security Token Service (AWS STS) untuk membuat kredensial keamanan sementara guna menandatangani permintaan.

Anda dapat memanggil operasi API ini dari lokasi jaringan mana pun, dan Anda dapat menggunakannyakebijakan akses berbasis sumber dayadiEventBridge, yang dapat mencakup pembatasan berdasarkan alamat IP sumber. Anda juga dapat menggunakan kebijakan EventBridge untuk mengontrol akses dari titik akhir Amazon Virtual Private Cloud (Amazon VPC) tertentu atau VPC tertentu. Secara efektif, ini mengisolasi akses jaringan ke sumber daya EventBridge yang diberikan hanya dari VPC tertentu dalam jaringan AWS.

Analisis konfigurasi dan kelemahan di Amazon EventBridge (Amazon EventBridge)

Konfigurasi dan kontrol IT merupakan tanggung jawab bersama antara AWS dan Anda, pelanggan kami. Untuk informasi selengkapnya, lihat model tanggung jawab bersama AWS.

Memantau Amazon EventBridge

EventBridge mengirimkan metrik ke Amazon CloudWatch setiap menit untuk semuanya, mulai dari jumlah peristiwa yang cocok hingga berapa kali target dipanggil oleh aturan.

Video berikut meninjau EventBridge perilaku pemantauan dan audit melalui CloudWatch: Pemantauan dan audit peristiwa

Topik

- EventBridge metrik
- Dimensi untuk EventBridge metrik

EventBridge metrik

Namespace `AWS/Events` mencakup metrik berikut.

Untuk metrik yang menggunakan Count sebagai satuan, Jumlahkan dan SampleCount cenderung menjadi statistik yang paling berguna.

Metrik yang hanya menentukan `RuleName` dimensi merujuk ke bus acara default. Metrik yang menentukan `RuleName` dimensi `EventBusName` dan dimensi mengacu pada bus acara khusus.

Metrik	Deskripsi
`DeadLetterInvocations`	Frekuensi target aturan tidak dipanggil sebagai tanggapan suatu peristiwa. Termasuk permohonan yang menjalankan kembali aturan yang sama, menyebabkan putaran tidak terbatas. Dimensi yang Valid: RuleName Unit: Hitungan
`Events`	Jumlah acara mitra yang dicerna oleh EventBridge. Dimensi yang Valid: EventSourceName Unit: Count

Metrik	Deskripsi
`FailedInvocations`	Jumlah permohonan yang gagal secara permanen. Tidak termasuk permohonan yang diulang atau berhasil setelah diupayakan lagi. Hal ini juga tidak menganggap permohonan gagal yang dihitung dalam `DeadLetterInvocations` . **Note** EventBridge hanya mengirimkan metrik ini ke CloudWatch jika bukan nol. Dimensi yang Valid: RuleName Unit: Hitungan
`Invocations`	Frekuensi target dipanggil oleh aturan dalam menanggapi suatu peristiwa. Termasuk permohonan yang berhasil dan gagal, tetapi tidak termasuk upaya yang dipaksakan atau dicoba lagi hingga gagal secara permanen. Itu tidak termasuk`DeadLetterInvocations` . **Note** EventBridge hanya mengirimkan metrik ini ke CloudWatch jika bukan nol. Dimensi yang Valid: Tidak ada, RuleName Unit: Hitungan
`InvocationAttempts`	Beberapa kali EventBridge mencoba memanggil target. Dimensi valid: Tidak Ada Unit: Hitungan

Metrik	Deskripsi
`InvocationsCreated`	Jumlah total pemanggilan yang dibuat sebagai tanggapan terhadap setiap acara. Metrik ini sering digunakan untuk memantau pemanfaatan batas throttle Invocations dalam kuota layanan transaksi per detik. EventBridge Dimensi valid: Tidak Ada Unit: Hitungan
`InvocationsFailedToBeSentToDlq`	Jumlah permohonan yang tidak dapat dipindahkan ke antrean surat mati. Kesalahan antrean surat mati dapat terjadi karena kesalahan izin, sumber daya tidak tersedia, atau batas ukuran. **Note** EventBridge hanya mengirimkan metrik ini ke CloudWatch jika bukan nol. Dimensi yang Valid: RuleName Unit: Hitungan
`IngestiontoInvocationCompleteLatency`	Waktu yang diambil dari konsumsi acara hingga penyelesaian upaya doa pertama yang berhasil. Dimensi yang Valid: EventBusName, Tidak Ada, RuleName Unit: Milidetik
`IngestiontoInvocationStartLatency`	Waktu untuk memproses peristiwa, diukur dari saat suatu peristiwa dicerna EventBridge hingga pemanggilan target pertama. Dimensi yang Valid: EventBusName, Tidak Ada, RuleName Unit: Milidetik

Metrik	Deskripsi
`InvocationsSentToDlq`	Jumlah permohonan yang dipindahkan ke antrean surat mati. **Note** EventBridge hanya mengirimkan metrik ini ke CloudWatch jika bukan nol. Dimensi yang Valid: RuleName Unit: Hitungan
`MatchedEvents`	Jika EventBusName atau EventSourceName ditentukan, jumlah peristiwa yang cocok dengan aturan apa pun. Jika RuleName ditentukan, jumlah peristiwa yang cocok dengan aturan tertentu. Dimensi yang Valid: EventBusName, RuleName, EventSourceName Unit: Hitungan
`RetryInvocationAttempts`	Berapa kali pemanggilan target telah dicoba lagi. **Note** EventBridge hanya mengirimkan metrik ini ke CloudWatch jika bukan nol. Dimensi valid: Tidak Ada Unit: Hitungan
`SuccessfulInvocationAttempts`	Berapa kali target berhasil dipanggil. Dimensi valid: Tidak Ada Unit: Hitungan

Metrik	Deskripsi
`ThrottledRules`	Berapa kali eksekusi aturan dibatasi. Doa untuk aturan tersebut mungkin tertunda. Untuk informasi selengkapnya, lihat Batas throttle Invocations dalam transaksi per detik di. ??? Dimensi yang Valid: EventBusName, Tidak Ada, RuleName Unit: Hitungan
`TriggeredRules`	Jumlah aturan yang telah berjalan dan sesuai dengan peristiwa apa pun. Anda tidak akan melihat metrik ini CloudWatch sampai aturan dipicu. Dimensi yang Valid: EventBusName, Tidak Ada, RuleName Unit: Hitungan

EventBridge PutEvents metrik

`AWS/Events`Namespace menyertakan metrik berikut yang berkaitan dengan permintaan API. `PutEvents`

Untuk metrik yang menggunakan Count sebagai satuan, Jumlahkan dan SampleCount cenderung menjadi statistik yang paling berguna.

Metrik	Deskripsi
`PutEvents ApproximateCallCount`	Perkiraan jumlah `PutEvents` permintaan yang diterima. Dimensi valid: Tidak Ada Unit: Hitungan
`PutEvents ApproximateFailedCount`	Perkiraan jumlah `PutEvents` permintaan yang gagal. Dimensi valid: Tidak Ada

Metrik	Deskripsi
	Unit: Hitungan
PutEvents ApproximateSuccessCount	Perkiraan jumlah PutEvents permintaan yang berhasil. Dimensi valid: Tidak Ada Unit: Hitungan
PutEvents ApproximateThrottledCount	Jumlah PutEvents permintaan ditolak karena pembatasan. Dimensi valid: Tidak Ada Unit: Hitungan
PutEvents EntriesCount	Jumlah entri acara yang terkandung dalam PutEvents permintaan. Dimensi valid: Tidak Ada Unit: Hitungan
PutEvents FailedEntriesCount	Jumlah entri acara yang terkandung dalam PutEvents permintaan yang gagal dicerna. Dimensi valid: Tidak Ada Unit: Hitungan
PutEvents Latency	Waktu yang dibutuhkan per PutEvents permintaan. Dimensi valid: Tidak Ada Unit: Milidetik
PutEvents RequestSize	Ukuran PutEvents permintaan. Dimensi valid: Tidak Ada Unit: Byte

EventBridge PutPartnerEvents metrik

`AWS/Events`Namespace menyertakan metrik berikut yang berkaitan dengan permintaan API. `PutPartnerEvents`

> **Note**
>
> EventBridge hanya menyertakan metrik yang berkaitan dengan PutPartnerEventspermintaan di akun mitra SaaS yang mengirim acara. Lihat informasi yang lebih lengkap di ???

Untuk metrik yang menggunakan Count sebagai satuan, Jumlahkan dan SampleCount cenderung menjadi statistik yang paling berguna.

Metrik	Deskripsi
`PutPartnerEventsApproximateCallCount`	Perkiraan jumlah `PutPartnerEvents` permintaan yang diterima. Dimensi valid: Tidak Ada Unit: Hitungan
`PutPartnerEventsApproximateFailedCount`	Perkiraan jumlah `PutPartnerEvents` permintaan yang gagal. Dimensi valid: Tidak Ada Unit: Hitungan
`PutPartnerEventsApproximateThrottledCount`	Jumlah `PutPartnerEvents` permintaan ditolak karena pembatasan. Dimensi valid: Tidak Ada Unit: Hitungan
`PutPartnerEventsApproximateSuccessCount`	Perkiraan jumlah `PutPartnerEvents` permintaan yang berhasil. Dimensi valid: Tidak Ada Unit: Hitungan

Metrik	Deskripsi
PutPartnerEventsEntriesCount	Jumlah entri acara yang terkandung dalam PutPartnerEvents permintaan. Dimensi valid: Tidak Ada Unit: Hitungan
PutPartnerEventsFailedEntriesCount	Jumlah entri acara yang terkandung dalam PutPartnerEvents permintaan yang gagal dicerna. Dimensi valid: Tidak Ada Unit: Hitungan
PutPartnerEventsLatency	Waktu yang dibutuhkan per PutPartnerEvents permintaan. Dimensi valid: Tidak Ada Unit: Milidetik

Dimensi untuk EventBridge metrik

EventBridge metrik memiliki dimensi, atau atribut yang dapat diurutkan, yang tercantum di bawah ini.

Dimensi	Deskripsi
EventBusName	Memfilter metrik yang tersedia berdasarkan nama bus acara.
EventSourceName	Memfilter metrik yang tersedia berdasarkan nama sumber acara mitra.
RuleName	Menyaring metrik yang tersedia berdasarkan nama aturan.

Memecahkan Masalah Amazon EventBridge

Anda dapat menggunakan langkah-langkah di bagian ini untuk memecahkan masalah Amazon. EventBridge

Topik

- Aturan saya berjalan tetapi fungsi Lambda saya tidak dipanggil
- Saya baru saja membuat atau memodifikasi aturan, tetapi tidak cocok dengan kejadian pengujian
- Aturan saya tidak berjalan pada waktu yang saya tentukan di ScheduleExpression
- Aturan saya tidak berjalan pada waktu yang saya harapkan
- Aturan saya cocok dengan panggilan API layanan AWS global tetapi tidak berjalan
- IAM role (IAM role) yang terkait dengan aturan saya diabaikan saat aturan berjalan
- Aturan saya memiliki pola kejadian yang seharusnya cocok dengan sumber daya, tapi tidak ada kejadian yang cocok
- Pengiriman kejadian saya ke target tertunda
- Beberapa kejadian tidak pernah dikirimkan ke target saya
- Aturan saya berjalan lebih dari sekali dalam menanggapi satu kejadian
- Mencegah loop tak terbatas
- Kejadian saya tidak dikirim ke antrean Amazon SQS target
- Aturan saya berjalan, tapi saya tidak melihat pesan yang diterbitkan ke topik Amazon SNS saya
- Topik Amazon SNS saya masih memiliki izin EventBridge bahkan setelah saya menghapus aturan yang terkait dengan topik Amazon SNS
- Kunci kondisi IAM mana yang dapat saya gunakan? EventBridge
- Bagaimana saya bisa tahu kapan EventBridge aturan dilanggar?

Aturan saya berjalan tetapi fungsi Lambda saya tidak dipanggil

Salah satu alasan mengapa fungsi Lambda Anda tidak berjalan adalah Anda tidak memiliki izin yang tepat.

Memeriksa izin Anda untuk fungsi Lambda

1. Menggunakan AWS CLI, jalankan perintah berikut dengan fungsi Anda dan AWS Wilayah Anda:

```
aws lambda get-policy --function-name MyFunction --region us-east-1
```

Anda akan melihat output berikut.

```
{
    "Policy": "{\"Version\":\"2012-10-17\",
    \"Statement\":[
        {\"Condition\":{\"ArnLike\":{\"AWS:SourceArn\":\"arn:aws:events:us-
east-1:123456789012:rule/MyRule\"}},
        \"Action\":\"lambda:InvokeFunction\",
        \"Resource\":\"arn:aws:lambda:us-east-1:123456789012:function:MyFunction\",
        \"Effect\":\"Allow\",
        \"Principal\":{\"Service\":\"events.amazonaws.com\"},
        \"Sid\":\"MyId\"}
    ],
    \"Id\":\"default\"}"
}
```

2. Jika Anda melihat pesan berikut.

```
A client error (ResourceNotFoundException) occurred when calling the GetPolicy
 operation: The resource you requested does not exist.
```

Atau, Anda melihat output tetapi Anda tidak dapat menemukan events.amazonaws.com sebagai entitas terpercaya dalam kebijakan, jalankan perintah berikut:

```
aws lambda add-permission \
--function-name MyFunction \
--statement-id MyId \
--action 'lambda:InvokeFunction' \
--principal events.amazonaws.com \
--source-arn arn:aws:events:us-east-1:123456789012:rule/MyRule
```

3. Jika output berisi bidang `SourceAccount`, maka Anda harus menghapusnya. `SourceAccount`Pengaturan EventBridge mencegah agar tidak dapat menjalankan fungsi.

> **Note**
>
> Jika kebijakan salah, Anda dapat mengedit aturan di EventBridge konsol dengan menghapus dan kemudian menambahkannya kembali ke aturan. EventBridge Konsol kemudian menetapkan izin yang benar pada target.
> Jika Anda menggunakan alias atau versi Lambda tertentu, tambahkan parameter `--qualifier` di perintah `aws lambda get-policy` dan `aws lambda add-permission`, seperti yang ditunjukkan dalam perintah berikut
>
> ```
> aws lambda add-permission \
> --function-name MyFunction \
> --statement-id MyId \
> --action 'lambda:InvokeFunction' \
> --principal events.amazonaws.com \
> --source-arn arn:aws:events:us-east-1:123456789012:rule/MyRule
> --qualifier alias or version
> ```

Saya baru saja membuat atau memodifikasi aturan, tetapi tidak cocok dengan kejadian pengujian

Saat Anda membuat perubahan aturan atau perubahan targetnya, kejadian yang masuk mungkin tidak langsung mulai atau berhenti cocok dengan aturan baru atau yang diperbarui. Tunggu sebentar hingga perubahan diterapkan.

Jika peristiwa masih tidak cocok setelah periode waktu yang singkat, periksa CloudWatch metrik `TriggeredRulesInvocations`, dan `FailedInvocations` untuk aturan Anda. Untuk informasi selengkapnya tentang metrik ini, lihat Memantau Amazon EventBridge.

Jika aturan dimaksudkan untuk mencocokkan acara dari AWS layanan, lakukan salah satu dari hal-hal ini:

- Gunakan `TestEventPattern` tindakan untuk menguji pola acara aturan Anda cocok dengan peristiwa pengujian. Untuk informasi selengkapnya, lihat TestEventPatterndi Referensi Amazon EventBridge API.
- Gunakan Sandbox di EventBridge konsol.

Aturan saya tidak berjalan pada waktu yang saya tentukan di `ScheduleExpression`

Pastikan Anda telah mengatur jadwal untuk aturan di zona waktu UTC+0. Jika `ScheduleExpression` benar, ikuti langkah-langkah di Saya baru saja membuat atau memodifikasi aturan, tetapi tidak cocok dengan kejadian pengujian.

Aturan saya tidak berjalan pada waktu yang saya harapkan

EventBridge menjalankan aturan dalam satu menit dari waktu mulai yang Anda tetapkan. Hitung mundur untuk waktu aktif dimulai segera setelah Anda membuat aturan.

> **Note**
>
> Aturan terjadwal memiliki jenis pengiriman `guaranteed` makna peristiwa akan dipicu untuk setiap waktu yang diharapkan setidaknya sekali.

Anda dapat menggunakan ekspresi cron untuk memanggil target pada waktu tertentu. Untuk membuat aturan yang berjalan setiap empat jam di menit ke-0, lakukan salah satu dari langkah berikut:

- Di EventBridge konsol, Anda menggunakan ekspresi `0 0/4 * * ? *` cron.
- Dengan menggunakan AWS CLI, Anda menggunakan ekspresi`cron(0 0/4 * * ? *)`.

Misalnya, untuk membuat aturan bernama `TestRule` yang berjalan setiap 4 jam dengan menggunakan AWS CLI, Anda menggunakan perintah berikut.

```
aws events put-rule --name TestRule --schedule-expression 'cron(0 0/4 * * ? *)'
```

Untuk menjalankan aturan setiap lima menit, gunakan ekspresi cron berikut.

```
aws events put-rule --name TestRule --schedule-expression 'cron(0/5 * * * ? *)'
```

Resolusi terbaik untuk EventBridge aturan yang menggunakan ekspresi cron adalah satu menit. Aturan terjadwal Anda berjalan di menit itu tetapi tidak tepat pada detik ke-0.

Karena EventBridge dan layanan target didistribusikan, mungkin ada penundaan beberapa detik antara waktu aturan terjadwal berjalan dan waktu layanan target melakukan tindakan pada sumber daya target.

Aturan saya cocok dengan panggilan API layanan AWS global tetapi tidak berjalan

AWS layanan global; seperti, IAM dan Amazon Route 53 hanya tersedia di Wilayah AS Timur (Virginia N.), sehingga acara dari panggilan AWS API dari layanan global hanya tersedia di wilayah tersebut. Untuk informasi selengkapnya, lihat Acara dari AWS layanan.

IAM role (IAM role) yang terkait dengan aturan saya diabaikan saat aturan berjalan

EventBridge hanya menggunakan peran IAM untuk aturan yang mengirim peristiwa ke aliran Kinesis. Untuk aturan yang memanggil fungsi Lambda atau topik Amazon SNS, Anda perlu memberikan Izin berbasis sumber daya.

Pastikan AWS STS titik akhir regional Anda diaktifkan, sehingga EventBridge dapat menggunakannya saat mengasumsikan peran IAM yang Anda berikan. Untuk informasi selengkapnya, lihat Mengaktifkan dan Menonaktifkan AWS STS di AWS Wilayah di Panduan Pengguna IAM.

Aturan saya memiliki pola kejadian yang seharusnya cocok dengan sumber daya, tapi tidak ada kejadian yang cocok

Sebagian besar layanan dalam AWS memperlakukan titik dua (:) atau garis miring (/) sebagai karakter yang sama di Amazon Resource Names (ARN)., tetapi EventBridge menggunakan kecocokan persis dalam pola dan aturan acara. Pastikan untuk menggunakan karakter ARN yang benar saat membuat pola kejadian sehingga mereka cocok dengan sintaks ARN di kejadian yang dicocokkan.

Beberapa peristiwa, seperti peristiwa panggilan AWS API dari CloudTrail, tidak memiliki apa pun di bidang sumber daya.

Pengiriman kejadian saya ke target tertunda

EventBridge mencoba mengirimkan acara ke target hingga 24 jam, kecuali dalam skenario di mana sumber daya target Anda dibatasi. Upaya pertama dilakukan segera setelah kejadian tiba di alur kejadian. Jika layanan target mengalami masalah, EventBridge secara otomatis menjadwal ulang pengiriman lain. Jika 24 jam telah berlalu sejak kedatangan acara, EventBridge berhentilah mencoba menyampaikan acara dan menerbitkan `FailedInvocations` metrik di CloudWatch. Kami menyarankan Anda menyiapkan DLQ untuk menyimpan acara yang tidak berhasil dikirim ke target. Lihat informasi yang lebih lengkap di Kebijakan coba lagi peristiwa dan menggunakan antrean surat mati

Beberapa kejadian tidak pernah dikirimkan ke target saya

Jika target EventBridge aturan dibatasi untuk waktu yang lama, EventBridge mungkin tidak mencoba lagi pengiriman. Misalnya, jika target tidak disediakan untuk menangani lalu lintas peristiwa yang masuk dan layanan target membatasi permintaan yang EventBridge dibuat atas nama Anda, maka EventBridge mungkin tidak mencoba lagi pengiriman.

Aturan saya berjalan lebih dari sekali dalam menanggapi satu kejadian

Dalam kasus yang jarang terjadi, aturan yang sama dapat berjalan lebih dari sekali untuk satu kejadian atau waktu yang dijadwalkan, atau target yang sama dapat dipanggil lebih dari sekali untuk sebuah aturan tertentu yang dipicu.

Mencegah loop tak terbatas

Dalam EventBridge, dimungkinkan untuk membuat aturan yang mengarah ke loop tak terbatas, di mana aturan berjalan berulang kali. Jika Anda memiliki aturan yang menyebabkan loop tak terbatas, tulis ulang aturan itu sehingga tindakan yang diambil aturan tersebut tidak cocok dengan aturan yang sama.

Misalnya, aturan yang mendeteksi bahwa ACL telah berubah di bucket Amazon S3 lalu menjalankan perangkat lunak untuk mengubahnya ke keadaan baru menyebabkan loop tak terbatas. Salah satu cara untuk mengatasinya adalah menulis ulang aturan sehingga aturan itu hanya cocok dengan ACL yang berada dalam keadaan buruk.

Loop tak terbatas dapat dengan cepat menyebabkan biaya yang lebih tinggi dari yang diharapkan. Kami merekomendasikan agar Anda menggunakan penganggaran, yang akan memberi tahu Anda bila biaya melampaui batas yang ditentukan. Untuk informasi selengkapnya, lihat Mengelola Biaya Anda dengan Anggaran.

Kejadian saya tidak dikirim ke antrean Amazon SQS target

Jika antrean Amazon SQS Anda dienkripsi, Anda harus membuat kunci KMS yang dikelola pelanggan dan menyertakan bagian izin berikut dalam kebijakan kunci KMS Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi AWS KMS izin.

```
{
  "Sid": "Allow EventBridge to use the key",
  "Effect": "Allow",
  "Principal": {
    "Service": "events.amazonaws.com"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
  ],
  "Resource": "*"
}
```

Aturan saya berjalan, tapi saya tidak melihat pesan yang diterbitkan ke topik Amazon SNS saya

Skenario 1

Anda perlu izin untuk menerbitkan pesan ke topik Amazon SNS Anda. Gunakan perintah berikut menggunakan AWS CLI, mengganti us-east-1 dengan Wilayah Anda dan menggunakan topik ARN Anda.

```
aws sns get-topic-attributes --region us-east-1 --topic-arn "arn:aws:sns:us-
east-1:123456789012:MyTopic"
```

Untuk memiliki izin yang benar, atribut kebijakan Anda mirip dengan yang berikut ini.

```
"{\"Version\":\"2012-10-17\",
```

```
\"Id\":\"__default_policy_ID\",
\"Statement\":[{\"Sid\":\"__default_statement_ID\",
\"Effect\":\"Allow\",
\"Principal\":{\"AWS\":\"*\"},
\"Action\":[\"SNS:Subscribe\",
\"SNS:ListSubscriptionsByTopic\",
\"SNS:DeleteTopic\",
\"SNS:GetTopicAttributes\",
\"SNS:Publish\",
\"SNS:RemovePermission\",
\"SNS:AddPermission\",
\"SNS:SetTopicAttributes\"],
\"Resource\":\"arn:aws:sns:us-east-1:123456789012:MyTopic\",
\"Condition\":{\"StringEquals\":{\"AWS:SourceOwner\":\"123456789012\"}}},{\"Sid\":
\"Allow_Publish_Events\",
\"Effect\":\"Allow\",
\"Principal\":{\"Service\":\"events.amazonaws.com\"},
\"Action\":\"sns:Publish\",
\"Resource\":\"arn:aws:sns:us-east-1:123456789012:MyTopic\"}]}"
```

Jika Anda tidak melihat `events.amazonaws.com` dengan izin `Publish` dalam kebijakan Anda, salin dahulu kebijakan saat ini dan tambahkan pernyataan berikut ke daftar pernyataan.

```
{\"Sid\":\"Allow_Publish_Events\",
\"Effect\":\"Allow\",\"Principal\":{\"Service\":\"events.amazonaws.com\"},
\"Action\":\"sns:Publish\",
\"Resource\":\"arn:aws:sns:us-east-1:123456789012:MyTopic\"}
```

Kemudian atur atribut topik dengan menggunakan AWS CLI, gunakan perintah berikut.

```
aws sns set-topic-attributes --region us-east-1 --topic-arn "arn:aws:sns:us-
east-1:123456789012:MyTopic" --attribute-name Policy --attribute-
value NEW_POLICY_STRING
```

> **Note**
>
> Jika kebijakan salah, Anda juga dapat mengedit aturan di EventBridge konsol dengan menghapus dan kemudian menambahkannya kembali ke aturan. EventBridge menetapkan izin yang benar pada target.

Skenario 2

Jika topik SNS Anda dienkripsi, Anda harus menyertakan bagian berikut dalam kebijakan kunci KMS Anda.

```
{
  "Sid": "Allow EventBridge to use the key",
  "Effect": "Allow",
  "Principal": {
    "Service": "events.amazonaws.com"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
  ],
  "Resource": "*"
}
```

Topik Amazon SNS saya masih memiliki izin EventBridge bahkan setelah saya menghapus aturan yang terkait dengan topik Amazon SNS

Saat Anda membuat aturan dengan Amazon SNS sebagai target, EventBridge tambahkan izin ke topik Amazon SNS Anda atas nama Anda. Jika Anda menghapus aturan segera setelah Anda membuatnya, EventBridge mungkin tidak menghapus izin dari topik Amazon SNS Anda. Jika ini terjadi, Anda dapat menghapus izin dari topik menggunakan perintah `aws sns set-topic-attributes`. Untuk informasi tentang izin berbasis sumber daya untuk mengirim kejadian, lihat Menggunakan kebijakan berbasis sumber daya untuk AmazonEventBridge.

Kunci kondisi IAM mana yang dapat saya gunakan? EventBridge

EventBridge mendukung tombol kondisi AWS-wide (lihat kunci konteks IAM dan AWS STS kondisi di Panduan Pengguna IAM), ditambah kunci yang tercantum di. Menggunakan ketentuan kebijakan IAM ketentuan untuk kontrol akses yang sangat baik

Bagaimana saya bisa tahu kapan EventBridge aturan dilanggar?

Anda dapat menggunakan alarm berikut untuk memberi tahu Anda ketika EventBridge aturan Anda dilanggar.

Membuat alarm untuk menginformasikan saat aturan rusak

1. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Pilih Buat Alarm. Di panel CloudWatch Metrik menurut Kategori, pilih Metrik Acara.
3. Dalam daftar metrik, pilih FailedInvocations.
4. Di atas grafik, pilih Statistik, Sum.
5. Untuk Periode, pilih satu nilai, misalnya 5 menit. Pilih Selanjutnya.
6. Di bawah Ambang Alarm, untuk Nama, ketikkan nama unik untuk alarm, misalnya myFailedRules. Untuk Deskripsi, masukkan deskripsi alarm, misalnya Aturan tidak mengirimkan acara ke target.
7. Untuk is, pilih >= dan 1. Untuk for, masukkan 10.
8. Di Tindakan, untuk Setiap kali alarm ini, pilih Status adalah ALARM.
9. Untuk Kirim notifikasi ke, pilih topik Amazon SNS yang sudah ada atau buat topik baru. Untuk membuat topik SNS baru, pilih Daftar baru. Ketik nama untuk topik Amazon SNS baru, misalnya:. myFailedRules
10. Untuk Daftar email, ketik daftar alamat email yang dipisahkan dengan tanda koma untuk menerima pemberitahuan ketika alarm berubah ke status ALARM.
11. Pilih Buat Alarm.

EventBridge Kuota Amazon

Ada kuota untuk sebagian besar aspek. EventBridge

Topik

- EventBridge kuota
- PutPartnerEvents kuota menurut Wilayah
- EventBridge Kuota Registri Skema
- EventBridge Kuota pipa

> **Note**
>
> Untuk daftar kuota EventBridge Penjadwal, lihat Kuota untuk Penjadwal di Panduan EventBridge Pengguna Penjadwal. EventBridge

EventBridge kuota

EventBridge memiliki kuota berikut.

Konsol Service Quotas memberikan informasi tentang EventBridge kuota. Sembari melihat kuota default, Anda dapat menggunakan konsol Service Quotas guna mengajukan penambahan kuota untuk kuota yang dapat disesuaikan.

Nama	Default	Dapat disesuaikan	Deskripsi
Tujuan API	Setiap Wilayah yang didukung: 3.000	Ya	Jumlah maksimum Tujuan API per akun per Wilayah.
Koneksi	Setiap Wilayah yang didukung: 3.000	Ya	Jumlah koneksi maksimal per akun per Wilayah.

Nama	Default	Dapat disesuaikan	Deskripsi
CreateEndpoint batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum permintaan per detik untuk CreateEndpoint API. Permintaan tambahan diperlambat.
DeleteEndpoint batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum permintaan per detik untuk DeleteEndpoint API. Permintaan tambahan diperlambat.
Titik akhir	Setiap Wilayah yang didukung: 100	Ya	Jumlah maksimum titik akhir per akun per Wilayah.
Ukuran kebijakan bus acara	Setiap Wilayah yang didukung: 10.240	Ya	Ukuran kebijakan maksimum, dalam karakter. Ukuran kebijakan ini meningkat setiap kali Anda memberikan akses ke akun lain. Anda dapat melihat kebijakan Anda saat ini dan ukurannya dengan menggunakan DescribeEventBus API.
Bus peristiwa	Setiap Wilayah yang didukung: 100	Ya	Bus acara maksimum per akun.

Nama	Default	Dapat disesuaikan	Deskripsi
Ukuran pola acara	Setiap Wilayah yang didukung: 2,048	Ya	Ukuran maksimum pola acara, dalam karakter.

Nama	Default	Dapat disesuaikan	Deskripsi
Batas throttle pemanggilan dalam transaksi per detik	us-timur-1:18.750 per detik us-timur-2:4.500 per detik us-barat-1:2.250 per detik us-barat-2:18.750 per detik af-selatan-1:750 per detik ap-timur laut-1:2.250 per detik ap-timur laut-3:750 per detik ap-tenggara 1:2.250 per detik ap-tenggara 2:2.250 per detik ap-tenggara 3:750 per detik eu-central-1:4.500 per detik eu-selatan-1:750 per detik	Ya	Pemanggilan adalah acara yang cocok dengan aturan dan dikirim ke target aturan. Setelah batas tercapai, pemanggilan dibatasi; yaitu, mereka masih terjadi tetapi mereka tertunda.

Nama	Default	Dapat disesuaikan	Deskripsi
	eu-barat-1:18.750 per detik eu-barat-2:2.250 per detik Masing-masing Daerah yang didukung lainnya: 1.100 per detik		
Jumlah aturan	af-selatan-1:100 eu-selatan-1:100 Masing-masing Wilayah yang didukung lainnya: 300	Ya	Jumlah maksimum aturan yang dapat dimiliki akun per bus acara

Nama	Default	Dapat disesuaikan	Deskripsi
PutEvents batas throttle dalam transaksi per detik	us-timur-1:10.000 per detik us-timur-2:2.400 per detik us-barat-1:1.200 per detik us-barat-2:10.000 per detik af-selatan-1:400 per detik ap-timur laut-1:1.200 per detik ap-timur laut-3:400 per detik ap-tenggara 1:1.200 per detik ap-tenggara 2:1.200 per detik ap-tenggara 3:400 per detik eu-central-1:2.400 per detik eu-selatan-1:400 per detik	Ya	Jumlah maksimum permintaan per detik untuk PutEvents API. Permintaan tambahan diperlambat.

Nama	Default	Dapat disesuaikan	Deskripsi
	eu-barat-1:10.000 per detik eu-barat-2:1.200 per detik Masing-masing Daerah yang didukung lainnya: 600 per detik		
Tingkat pemanggilan per tujuan API	Setiap Wilayah yang didukung: 300 per detik	Ya	Jumlah permintaan maksimal untuk dikirim ke setiap titik akhir tujuan API per akun per Wilayah. Setelah kuota terpenuhi, permintaan yang akan datang untuk titik akhir API dicekik. Permintaan masih akan berlaku, tetapi ditangguhkan.
Target per aturan	Setiap Wilayah yang didukung: 5	Tidak	Jumlah maksimum target yang dapat dikaitkan dengan aturan
Batas throttle PutEvents dalam transaksi per detik	Setiap Wilayah yang didukung: 50 per detik	Ya	Jumlah maksimum permintaan per detik untuk semua operasi EventBridge API kecuali PutEvents. Permintaan tambahan dibatasi

Nama	Default	Dapat disesuaikan	Deskripsi
UpdateEndpoint batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum permintaan per detik untuk UpdateEndpoint API. Permintaan tambahan diperlambat.

Selain itu, EventBridge memiliki kuota berikut yang tidak dikelola melalui konsol Service Quotas.

Nama	Default	Deskripsi
Bus peristiwa	Setiap Wilayah yang didukung: 100	Bus acara maksimum per akun.
Ukuran kebijakan bus acara	Setiap Wilayah yang didukung: 10240	Ukuran kebijakan maksimum, dalam karakter. Ukuran kebijakan ini meningkat setiap kali Anda memberikan akses ke akun lain. Anda dapat melihat kebijakan Anda saat ini dan ukurannya dengan menggunakan `DescribeEventBus` API.
Ukuran pola acara	Setiap Wilayah yang didukung: 2048	Ukuran maksimum pola acara, dalam karakter. Ini dapat disesuaikan hingga 4096 karakter. Jika Anda memiliki persyaratan untuk batas maksimum yang lebih tinggi, hubungi dukungan.
Aturan yang berisi wildcard	Setiap Wilayah yang didukung: 30	Jumlah maksimum aturan, per bus acara per akun, yang dapat berisi filter acara yang menyertakan wildcard. Kuota ini tidak dapat disesuaikan.

Nama	Default	Deskripsi
	aturan per bus acara	Untuk informasi selengkapnya tentang penggunaan wildcard dalam pola acara, lihat???.
Tingkat penemuan skema	Setiap Wilayah yang didukung: 255 level	Jumlah maksimum penemuan skema level akan menyimpulkan peristiwa yang bersarang. Setiap peristiwa melewati level 255 diabaikan.

PutPartnerEvents kuota menurut Wilayah

Jika Anda memiliki persyaratan untuk batas maksimum yang lebih tinggi, hubungi dukungan.

Wilayah	Transaksi per detik (TPS)
• AWS GovCloud (AS-Barat) • AWS GovCloud (AS-Timur) • AS Timur (N. Virginia) • AS Timur (Ohio) • AS Barat (California Utara) • AS Barat (Oregon) • Afrika (Cape Town) • Asia Pasifik (Hong Kong) • Asia Pasifik (Mumbai) • Asia Pasifik (Osaka) • Asia Pasifik (Seoul) • Asia Pasifik (Singapura) • Asia Pasifik (Sydney) • Asia Pasifik (Tokyo) • Kanada (Pusat) • Eropa (Frankfurt)	PutPartnerEventsmemiliki batas lunak 1.400 permintaan throughput per detik dan 3.600 permintaan burst per detik secara default di semua Wilayah.

Wilayah	Transaksi per detik (TPS)
• Eropa (Irlandia) • Eropa (London) • Eropa (Milan) • Eropa (Paris) • Europe (Stockholm) • Eropa (Milan) • Amerika Selatan (Sao Paulo) • Tiongkok (Ningxia) • Tiongkok (Beijing)	

EventBridge Kuota Registri Skema

EventBridge Schema Registry memiliki kuota berikut.

Konsol Service Quotas memberikan informasi tentang EventBridge kuota. Sembari melihat kuota default, Anda dapat menggunakan konsol Service Quotas guna mengajukan penambahan kuota untuk kuota yang dapat disesuaikan.

Nama	Default	Dapat disesuaikan	Deskripsi
DiscoveredSchemas	Setiap Wilayah yang didukung: 200	Ya	Jumlah maksimum skema untuk registri skema yang ditemukan yang dapat Anda buat di wilayah saat ini
Penemu	Setiap Wilayah yang didukung: 10	Ya	Jumlah maksimum penemu yang dapat Anda buat di wilayah saat ini.
Registrasi	Setiap Wilayah yang didukung: 10	Ya	Jumlah maksimum pendaftar yang dapat

Nama	Default	Dapat disesuaikan	Deskripsi
			Anda buat di wilayah saat ini.
SchemaVersions	Setiap Wilayah yang didukung: 100	Ya	Jumlah maksimum versi per skema yang dapat Anda buat di wilayah saat ini.
Skema	Setiap Wilayah yang didukung: 100	Ya	Jumlah maksimum skema per registri yang dapat Anda buat di wilayah saat ini. (Kecuali Registri Skema yang Ditemukan)

EventBridge Kuota pipa

EventBridge Pipa memiliki kuota berikut. Jika Anda memiliki persyaratan untuk batas maksimum yang lebih tinggi, hubungi dukungan.

Sumber daya	Wilayah	Batas default
Eksekusi pipa bersamaan per akun	• AWS GovCloud (AS-Barat) • AWS GovCloud (AS-Timur) • Tiongkok (Ningxia) • China (Beijing) • Asia Pacific (Osaka) • Africa (Cape Town) (Africa (Cape Town)) • Europe (Milan) • AS Timur (Ohio) • Eropa (Frankfurt)	1000

Sumber daya	Wilayah	Batas default
	• US West (N. California) • Eropa (London) • Asia Pacific (Sydney) • Asia Pacific (Tokyo) • Asia Pacific (Singapore) (Asia Pacific (Singapore)) • Canada (Central) • Eropa (Paris) • Eropa (Stockholm) • South America (São Paulo) • Asia Pacific (Seoul) • Asia Pacific (Mumbai) • Asia Pacific (Hong Kong) • Middle East (Bahrain) (Middle East (Bahrain)) • China (Ningxia) • China (Beijing) • Asia Pacific (Osaka) • Africa (Cape Town) (Africa (Cape Town)) • Europe (Milan)	
Eksekusi pipa bersamaan per akun	• AS Timur (Virginia Utara) • US West (Oregon) • Eropa (Irlandia)	3000
Pipa per akun	Semua	1000

EventBridge Tag Amazon

Tag adalah label atribut kustom yang Anda atau AWS tetapkan ke AWS sumber daya. Di EventBridge, Anda dapat menetapkan tag ke bus aturan dan acara. Setiap sumber daya dapat memiliki maksimum 50 tag.

Anda menggunakan tag untuk mengidentifikasi dan mengatur AWS sumber daya Anda. Banyak AWS layanan mendukung penandaan, sehingga Anda dapat menetapkan tag yang sama ke sumber daya dari layanan yang berbeda untuk menunjukkan bahwa sumber daya terkait. Misalnya, Anda dapat menetapkan tag yang sama ke EventBridge aturan yang Anda tetapkan ke instans EC2.

Setiap tanda memiliki dua bagian:

- Kunci tanda, misalnya, `CostCenter`, `Environment`, atau `Project`.
  - Kunci tanda peka terhadap huruf besar dan kecil.
  - Panjang kunci tanda maksimum adalah 128 karakter Unicode dalam UTF-8.
  - Untuk setiap sumber daya, setiap kunci tanda harus unik.
  - Karakter yang diperbolehkan adalah huruf, angka, spasi yang dapat mewakili dalam UTF-8, dan karakter berikut: . : + = @ _ / - (tanda penghubung).
  - `aws:`Awalan dilarang untuk tag karena dicadangkan untuk AWS digunakan. Anda tidak dapat menyunting atau menghapus kunci atau nilai tag dengan awalan ini. Tanda dengan prefiks ini tidak dihitung terhadap tanda Anda per batas sumber daya.
- Bidang nilai tag opsional, misalnya, `111122223333` atau `Production`.
  - Setiap kunci tanda hanya dapat memiliki satu nilai.
  - Kunci dan nilai tanda peka huruf besar dan kecil.
  - Mengabaikan nilai tanda sama dengan menggunakan rangkaian kosong.
  - Panjang nilai tanda maksimum adalah 256 karakter Unicode dalam UTF-8.
  - Karakter yang diperbolehkan adalah huruf, angka, spasi yang dapat mewakili dalam UTF-8, dan karakter berikut: . : + = @ _ / - (tanda penghubung).

> **Tip**
>
> Sebagai praktik terbaik, tentukan strategi untuk memanfaatkan tag dan secara konsisten menerapkan strategi tersebut di semua jenis sumber daya. Misalnya, putuskan apakah

akan menggunakan `Costcenter`, `costcenter`, atau `CostCenter` dan menggunakan kesepakatan yang sama untuk semua tag.

Anda dapat menggunakan EventBridge konsol, EventBridge API, atau AWS CLI untuk menambahkan, mengedit, atau menghapus tag. Untuk informasi selengkapnya, lihat hal berikut:

- TagResource, UntagResource, dan ListTagsForResourcedi Referensi EventBridge API Amazon
- tag-resource, untag-resource, dan di Referensi list-tags-for-resourceAWS CLI
- Bekerja dengan Editor Tanda dalam Panduan Pengguna Resource Groups

Riwayat Dokumen

Tabel berikut menjelaskan tentang perubahan penting dalam setiap rilis Panduan EventBridge Pengguna Amazon, yang dimulai pada Juli 2019. Untuk notifikasi tentang pembaruan dokumentasi ini, Anda dapat berlangganan ke umpan RSS.

Perubahan	Deskripsi	Tanggal Rilis
Menghasilkan AWS CloudFormation template dari bus acara dan aturan.	Anda sekarang dapat membuat AWS CloudFormation template dari bus dan aturan EventBridge acara Amazon yang ada. • Buat AWS CloudFormation template dari bus EventBridge acara Amazon	18 November 2022
Meluncurkan dokumentasi EventBridge Pipa.	Anda sekarang dapat membuat pipa untuk menghubungkan sumber ke target, dengan penyaringan dan pengayaan opsional. • Pipa	1 Desember 2022
Menghasilkan AWS CloudFormation template dari bus acara dan aturan.	Anda sekarang dapat membuat AWS CloudFormation template dari bus dan aturan EventBridge acara Amazon yang ada. • Buat AWS CloudFormation template dari bus EventBridge acara Amazon	18 November 2022
Ditambahkan AmazonEventBridgePipesFullAccess kebijakan.	Menyediakan akses penuh ke Amazon EventBridge Pipes. • EventBridge Kebijakan terkelola khusus pipa	1 Desember 2022
Ditambahkan AmazonEventBridgeP	Menyediakan akses baca saja ke Amazon EventBridge Pipes. • EventBridge Kebijakan terkelola khusus pipa	1 Desember 2022

Perubahan	Deskripsi	Tanggal Rilis
ipesReadOnlyAccess kebijakan.		
Ditambahkan AmazonEventBridgePipesOperatorAccess kebijakan.	Menyediakan akses read-only dan operator (yaitu, kemampuan untuk menghentikan dan mulai menjalankan Pipes) ke Amazon EventBridge Pipes. • EventBridge Kebijakan terkelola khusus pipa	1 Desember 2022
Memperbarui CloudWatchEventsFullAccess kebijakan.	Diperbarui untuk mencocokkan`AmazonEventBridgeFullAccess` . • AmazonEventBridgeFullAccess kebijakan	1 Desember 2022
Memperbarui CloudWatchEventsReadOnlyAccess kebijakan.	Diperbarui untuk mencocokkan`AmazonEventBridgeReadOnlyAccess` . • AmazonEventBridgeReadOnlyAccess kebijakan	1 Desember 2022
Diperbarui penyaringan konten dalam pola acara.	Anda sekarang dapat menggunak an`suffix`,`equals-ignore-case` , dan opsi`$or` penyaringan untuk membuat pola acara. • Pemfilteran konten dalam pola EventBridge acara Amazon	14 November 2022
Memperbarui AmazonEventBridgeFullAccess kebijakan.	Izin tambahan yang diperlukan untuk menggunak an EventBridge Schema Registry dan EventBridge Scheduler. • AmazonEventBridgeFullAccess kebijakan	10 November 2022

Perubahan	Deskripsi	Tanggal Rilis
Memperbarui AmazonEventBridgeReadOnlyAccess kebijakan.	Anda sekarang dapat melihat informasi EventBridge Schema Registry dan EventBridge Scheduler. • AmazonEventBridgeReadOnlyAccess kebijakan	10 November 2022
Diperbarui penyaringan konten dalam pola acara.	Anda sekarang dapat menggunak an`suffix`,`equals-ignore-case` , dan opsi`$or` penyaringan untuk membuat pola acara. • Pemfilteran konten dalam pola EventBridge acara Amazon	14 November 2022
Memperbarui AmazonEventBridgeFullAccess kebijakan.	Izin tambahan yang diperlukan untuk menggunak an EventBridge Schema Registry dan EventBridge Scheduler. • AmazonEventBridgeFullAccess kebijakan	10 November 2022
Memperbarui AmazonEventBridgeReadOnlyAccess kebijakan.	Anda sekarang dapat melihat informasi EventBridge Schema Registry dan EventBridge Scheduler. • AmazonEventBridgeReadOnlyAccess kebijakan	10 November 2022
Memperbarui AmazonEventBridgeReadOnlyAccess kebijakan.	Anda sekarang dapat melihat informasi endpoint. • AmazonEventBridgeReadOnlyAccess kebijakan	Selasa, 07 April 2022

Perubahan	Deskripsi	Tanggal Rilis
Tambahan dukungan untuk endpoint global.	Amazon EventBridge sekarang mendukung penggunaan endpoint global untuk membantu membuat aplikasi Anda toleran terhadap kesalahan regional tanpa biaya tambahan. Untuk pelajari selengkapnya, lihat di bawah ini: • Membuat aplikasi toleransi patahan daerah dengan endpoint global dan replikasi peristiwa • `CreateEndpoint`	Selasa, 07 April 2022
Dukungan tambahan untuk arsip dan pemutaran ulang peristiwa.	Amazon EventBridge sekarang mendukung penggunaan arsip untuk menyimpan peristiwa, dan memutar ulang peristiwa dari arsip. Untuk pelajari selengkapnya, lihat di bawah ini: • Mengarsipkan Amazon EventBridge acara. • `CreateArchive` • `StartReplay`	05 November 2020
Dukungan tambahan untuk antrian surat mati dan kebijakan percobaan kembali untuk target.	Amazon EventBridge sekarang mendukung penggunaan antrian surat mati dan mendefinisikan kebijakan percobaan kembali untuk target. Untuk pelajari selengkapnya, lihat di bawah ini: • Kebijakan coba lagi peristiwa dan menggunakan antrean surat mati. • `PutTargets`	12 Oktober 2020

Perubahan	Deskripsi	Tanggal Rilis
Dukungan tambahan untuk skema format JsonSchema Draft4.	Amazon EventBridge sekarang mendukung skema dalam format JsonSchema Draft 4. Sekarang Anda juga dapat mengekspor skema menggunakan EventBridge API. Untuk mempelajari selengkap nya, lihat berikut. • EventBridge Skema Amazon • `Export`dalam EventBridge Skema Registry API Referensi.	Senin, 28 September 2020
Kebijakan berbasis sumber daya untuk Pendaftaran EventBridge Skema	Sekarang pendaftaran EventBridge Skema Amazon Skema Amazon mendukung kebijakan berbasis sumber daya. Untuk informasi selengkap nya, lihat berikut ini. • Kebijakan berbasis sumber daya untuk skema Amazon EventBridge • `Policy`dalam EventBridge Skema Registry API Referensi • RegistryPolicy Jenis Sumber Daya dalam PanduanAWS CloudFormation Pengguna	30 April 2020
Tag untuk Bus Peristiwa	Perilisan ini memungkinkan Anda untuk membuat dan mengelola tag untuk bus peristiwa. Anda dapat menambahkan tag saat membuat bus peristiwa, dan menambahkan atau mengelola tag yang ada dengan memanggil API terkait. Untuk informasi lebih lanjut, lihat di bawah ini. • EventBridge Tag Amazon • Kebijakan Berbasis Tanda • `TagResource` • `UntagResource` • `ListTagsForResource`	24 Februari 2020

Perubahan	Deskripsi	Tanggal Rilis
Service Quotas ditambahkan	Amazon EventBridge telah memperbanyak kuota untuk permohonan dan untuk`PutEvents` . Kuota bervariasi menurut wilayah, dan dapat diperbanyak jika perlu.	11 Februari 2020
Menambahkan topik baru tentang mengubah input target, dan menambahkan link ke Application Auto Scaling Events.	Memperbaiki dokumentasi pada transformator input. • Transformasi EventBridge masukan Amazon • Gunakan Transformator Input untuk mengekstrasi data dari suatu peristiwa dan masukkan data tersebut ke target • Tutorial: Gunakan transformator input untuk menyesuaikan apa yang EventBridge lolos ke target peristiwa Menambahkan link ke Application Auto Scaling Events. • Application Auto Scaling Events dan EventBridge • Acara dari AWS layanan	20 Desember 2019
Penyaringan berbasis konten		19 Desember 2019
Menambahkan link ke contoh peristiwa Amazon Augmented AI.	Menambahkan tautan ke topik Amazon Augmented AI di Panduan Amazon SageMaker Developer yang menyediakan contoh peristiwa untuk Amazon Augmented AI. Untuk informasi selengkapnya, lihat berikut ini. • Gunakan Peristiwa di Amazon Augmented AI • Acara dari AWS layanan	13 Desember 2019

Perubahan	Deskripsi	Tanggal Rilis
Menambahkan link ke contoh peristiwa Amazon Chime.	Menambahkan link ke topik Amazon Chime yang menyediakan contoh peristiwa untuk layanan tersebut. Untuk informasi lebih lanjut, lihat di bawah ini. • Mengotomatisasi Amazon Chime dengan EventBridge • Acara dari AWS layanan	12 Desember 2019
EventBridge Skema Amazon	Sekarang Anda dapat mengelola skema dan menghasilkan binding kode untuk peristiwa di Amazon EventBridge. Untuk informasi selengkapnya, lihat berikut ini. • EventBridge Skema Amazon • EventBridge Skema Referensi API • EventSchemas Referensi Jenis Sumber Daya diAWS CloudFormation	1 Desember 2019
AWS CloudFormation dukungan untuk Bus Peristiwa	AWS CloudFormationsekarang mendukung sumber EventBus daya. Juga mendukung EventBusName parameter di kedua sumber daya EventBusPolicy dan Aturan. Untuk informasi selengkapnya, lihat Referensi Tipe EventBridge Sumber Daya Amazon.	7 Oktober 2019
Layanan baru	Rilis awal Amazon EventBridge.	11 Juli 2019

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